Data replication strategies with performance objective in data grid systems: a survey

Riad Mokadem* and Abdelkader Hameurlain

Institut de Recherche en Informatique de Toulouse (IRIT),
Paul Sabatier University,
118 Route de Narbonne 31062, Toulouse, France
Email: mokadem@irit.fr
Email: hameurlain@irit.fr
*Corresponding author

Abstract: Replicating for performance constitutes an important issue in large-scale data management systems. In this context, a significant number of replication strategies have been proposed for data grid systems. Some works classified these strategies into static vs. dynamic or centralised vs. decentralised or client vs. server initiated strategies. Very few works deal with a replication strategy classification based on the role of these strategies when building a replica management system. In this paper, we propose a new replication strategy classification based on objective functions of these strategies. Also, each replication strategy is designed according to the data grid topology for which it was proposed. We point out the impact of the topology on replication performance although most of these strategies have been proposed for a hierarchical grid topology. We also study the impact of some factors on performance of these strategies, e.g. access pattern, bandwidth consumption and storage capacity.

Keywords: data grid systems; data replication; replication strategies; classification; objective functions; performance.

Reference to this paper should be made as follows: Mokadem, R. and Hameurlain, A. (2015) 'Data replication strategies with performance objective in data grid systems: a survey', *Int. J. Grid and Utility Computing*, Vol. 6, No. 1, pp.30–46.

Biographical notes: Riad Mokadem is currently an Associate Professor in Computer Science at Paul Sabatier University, Toulouse, France, and a member of the IRIT Laboratory. His main research interests are query optimisation in large-scale distributed environments, scalable, distributed data structures and database performance.

Abdelkader Hameurlain is a Full Professor in Computer Science at Paul Sabatier University (IRIT Laboratory) Toulouse, France. His current research interests are in query optimisation in parallel and large-scale distributed environments, and mobile databases. He has been the general chair of the International Conference on Database and Expert Systems Applications (DEXA'02). He is co-editors in Chief of the international journal '*Transactions on Large-Scale Data- and Knowledge-Centered Systems*' (LNCS, Springer). He was guest editor of two special issues of *International Journal of Computer Systems Science and Engineering* on 'Mobile Databases' and 'Data Management in Grid and P2P Systems'.

1 Introduction

Today such as high-energy physics and bioinformatics applications produce a huge volume of data that may be accessed and shared at distributed nodes. This constitutes a good challenge regarding the access and processing of data in large-scale environments. In this context, data replication is an important optimisation method that deals with the generated problems. It consists of storing multiple copies of data, called replicas, at multiple nodes (Bernstein et al., 1987). Data replication has been commonly used in: (a) Database Management Systems (DBMS) (Perez et al., 2010), (b) parallel and distributed systems (Loukopoulos et al., 2005; Benoit and Rehn-Sonigo, 2008), (c) mobile systems (Tu et al., 2006) and (d) large-scale systems including P2P (Goel and Buyya, 2006; Xhafa et al., 2012a) and data grid systems (Ranganathan and Foster, 2001; Chervenak et al., 2002; Bell et al., 2003a; Lamehamedi et al., 2003; Abawajy, 2004; Park et al., 2004; Chang et al., 2006; Rahman et al., 2008; Rasool et al., 2009; Sashi and Thanamani, 2011; Abdullah et al., 2012; Mansouri and Dastghaibyfard, 2012; Devakirubai and Kannammal, 2013). In DBMS and distributed systems, replication designers pay attention to manage updates as well as performance of read-only queries. Although P2P systems are mostly designed for applications dealing with read-only queries, several other research works deal with transactional

queries. In data grid systems, most of the works in the literature deal with read-only queries. However, if the application has a read-only nature, replication can greatly improve the performance. But, if the application needs to process update queries, the benefits of replication can be neutralised by the overhead of maintaining consistency among multiple replicas. In consequence, an important global synchronisation with appropriate protocols is needed, i.e. many nodes may communicate with each other. Such protocols are generally not scalable. In consequence, if we apply replication as a scaling technique, we generally need to compromise on consistency (Van Steen and Pierre, 2010). In this paper, we mainly focus on the scenario dealing with the read-only queries, i.e. without any consistency managing, in order to achieve performance in data grid systems. We defer other issues to future work.

Data replication aims to keep the data close to the user where the query originated. It constitutes a common solution to (a) improve availability and reliability of data, (b) reduce the bandwidth consumption and (c) achieve fault tolerance by managing the departure/arrival of nodes in the system. An ideal solution to improve data availability is to replicate data in all nodes. Thus, data access cost will be significantly reduced. However, this solution is not realistic because of the storage and bandwidth constraints. Then, replication strategies are needed to determine *which* data is concerned by replication, *when* a replica should be created, *where* to place replicas (replica placement), *when* to remove replicas and *how* to locate the best replica (replica selection).

A significant number of replication strategies have been proposed in the literature. Most of them do not satisfy all the requirements cited above simultaneously. Furthermore, most of these strategies are designed for the hierarchical data grid topology. Throughout this paper, we point out advantages and disadvantages of all grid topologies (hierarchical, graph, P2P and hybrid) in order to that data replication can achieve performance. On the other hand, most of works in the literature have classified replication strategies according four aspects:

1 Static vs. dynamic replication strategies (Sashi and Thanamani, 2011; Khanli et al., 2011; Amjad et al., 2012). Although static strategies (Chervenak et al., 2002; Tatebe et al., 2002; Bell et al., 2003a) have advantages of having no overhead; the dynamic strategies (Lamehamedi et al., 2003; Chang et al., 2006; Rahman et al., 2008; Lee et al., 2011) are more suitable for data grid. In fact, this type of replication ensures that the benefits of replication will be continued even if user's behaviours as access pattern are changed, which corresponds to the dynamic properties of data grids.

2 Centralised vs. decentralised strategies (Sashi and Thanamani, 2011; Mansouri and Dastghaibyfard, 2012; Amjad et al., 2012). This classification concerns mainly the dynamic strategies that may be implemented either in a centralised or decentralised manner. In the first approach (Tang et al., 2005; Lei and Vrbsky, 2006; Chang and Chang, 2008; Lin et al., 2008; Bsoul et al., 2010), a replica central server is required to manage the replication process which conducts to extensive access latency and load on this server. On the other hand, some synchronisation is involved in order to provide better results in the decentralised approach (Ranganathan and Foster, 2001; Lei et al., 2008; Sashi and Thanamani, 2010).

3 Server vs. client initiated replication strategies (Dogan, 2009; Van Steen and Pierre, 2010). This classification relates to the origin of initiating replication. It can be client initiated (also called pull based) or server initiated (also called push based) replication. The server corresponds to the node that decides to make a replica and send it to other nodes when the client corresponds to the node which requests the data.

4 Unconditional vs. conditional replication strategies (Al Mistarihi and Yong, 2008). This classification deals with the nature of replication initiating, i.e. the replica-creation mechanism triggers according to some condition or not.

To the best of our knowledge, very few papers (Goel and Buyya, 2006) deal (partially) with a replication strategy classification based on the role of these strategies. In this paper, we propose a replication strategy classification in data grid systems regarding the different objective functions of these strategies. By using a given objective function, we define the role of the replication strategy that addresses separate issues when building a replica management system. In this context, we distinguish replication strategies based on (a) the popularity of data while exploiting temporal locality (Ranganathan and Foster, 2001), geographical locality (Nukarapu et al., 2011) and spatial locality (Khanli et al., 2011), (b) the network congestion (Sashi and Thanamani, 2011), (c) economic behaviours (Andronikou et al., 2012) and (d) cost models (Lamehamedi et al., 2003; Zhang et al., 2010). For each objective function, we describe the most important replication strategies and their main characteristics. A synthesis of the most important replication strategies is presented in order to point out their characteristics, e.g. the achieved function objective and their capability to achieve performance. Access cost, bandwidth consumption, access pattern and storage capacity are very important factors that impact on performance of these strategies. Some replication strategies deal with only a part of these factors. Hence, optimising some factors, e.g. access cost, and reducing the cost of replication may be conflicting goals. As an example, a frequent transfer of data in order to keep them close to the user can lead to strain on the network's resource. We enumerate existing trade-offs to advantage one factor to another. The simulation analysis permits us to enumerate the impact of some of these factors on the replication strategy performance. The impact of the data grid topology is also measured through a simulation based on the total mean job execution time metric.

The rest of this paper is organised as follows: Section 2 introduces replication strategies and their roles when replicating in data grid systems. Section 3 shows the impact of grid topology on replication strategies. Section 4 presents our replication strategy classification dealing with objective functions. Section 5 points out the important factors when data replication achieves performance. Section 6 presents the cost analysis of replication strategies. Section 7 deals with a simulation analysis that measures the impact of some factors on performance. Section 8 deals with the related work. Finally, Section 9 contains conclusion and future work.

2 Replication strategies

Managing a huge amount of data, spread on a large-scale network, constitutes an important challenge in data grid environments. In this context, replicating data at multiple nodes and then accessing them from the nearest node permit an efficient data access without a large consumption of bandwidth.

Replicating data in all nodes, which significantly reduce data access cost, is not realistic since this solution generates a large bandwidth consumption. Also, nodes have not always the capacity to store all these data. Dealing with these problems, distributing the replicas into data grid nodes is done according to replication strategies which answer the following questions:

Which replica is concerned by the creation/deletion? (The concerned data).

- *When* to create/delete replicas?
- *Where* to place new replicas? (Replica placement).
- *How* to select the best replica among many replicas available in the grid? (Replica selection).

Dealing with the above questions, there are four issues to be addressed by any dynamic replication strategy in order to achieve an optimised replication: (a) replica granularity that decides at which granularity we replicate the data, (b) replica creation/deletion, (c) replica placement which consists in placing the replicas on the appropriate node and (d) replica selection which is the process of choosing a replica from among those spreading across the data grid. All these issues that we describe in the next sub-sections should be beneficial with respect to several aspects:

1 Availability of data: when a fail occurs in any node, data replicated at another node can be used,

2 Reliability of data: an optimal number of replicas increase the probability that the query processing will be done completely. Hence, such a system is more reliable,

3 Scalability: replication strategies improve the scalability independently of the topology chosen for the data grid that we discuss in the following section,

4 Performance: performance results from different factor as the fact that data are close to the user (data locality), the decreasing on data access latency and the bandwidth consumption and,

5 Fault tolerance: some replication strategies deal with the dynamic properties of nodes that can join/leave the system at any moment.

2.1 Replica granularity

The granularity of a data replication corresponds to the unit of data that may be replicated independently of other units of data. Ideally, a replication strategy must adapt to any data granularity. However, replicating for performance requires deciding on data granularity since performance of replication strategies differs when dealing with different data units. In the literature, replication strategies are classified according to three levels of data sub-division: (a) individual files (Kunszt et al., 2005); (b) multiple files at the same time work, i.e. granularity of data sets (Garcia-Carballeira et al., 2007); and (c) smaller sub-divisions of files such as objects or fragments in order to save the storage space (Van Steen and Pierre, 2010). However, most of replication strategies we cited in this paper deal with the individual file granularity.

2.2 Replica placement

A naïve placement strategy may conduct to a system with some overloaded nodes and other nodes underutilised. In consequence, placing replicas in suitable nodes is preferable, e.g. the workload among replicas is balanced (Liu and Wu, 2006). A strategic placement has the objective of finding the optimal location for replicas, e.g. where the particular file has been often accessed (Mansouri and Dastghaibyfard, 2012). This improves the availability of data and speed up the data access. Recall that most of replica placement algorithms try to define the optimal number of replicas.

2.3 Replica selection

The process of selecting the best replica when different nodes hold replicas is called the replica selection. It aims to find the physical locations of multiple replicas from those copies geographically spreading in a large-scale system. Each grid node has its own capabilities and characteristics. Hence, choosing the appropriate replica from many replicas that have the required data is an important decision. This process is based on some characteristics that influence the response time as the data transfer time, the number of requests, the storage access latency and the distance between nodes (Sashi and Thanamani, 2010).

3 Impact of data grid topology on data replication strategies

Nodes under a replica management system can be organised into a variety of topologies. However, it has been proved

that scalability of a system is dependent upon the topology on which this system is based. In consequence, a replication strategy is designed according to the data grid topology for which this strategy is proposed.

Most of replication strategies in the literature have been proposed for the following topologies: hierarchical (e.g. multi-tier), peer-to-peer, hybrid and general graph topologies. In this section, we describe each data grid topology and give advantages/disadvantages of each of them, when a replication strategy is based on.

3.1 Hierarchical topology

This topology provides an efficient solution for sharing data, computational and network resources. Nodes are arranged in a tree-like hierarchy adopted in many scientific projects to support large-scale distributed computing. The multi-tier data grid is the most famous example of the hierarchical data grid. It can contain three or more tiers in the hierarchy. Each node belongs to a specific *tier*. The MONARC project (Monarc, see http://monarc.web.cern.ch/MONARC/) adopted a hierarchical network structure that has five tiers: the tier 0 is the main data source in which raw data are generated in CERN (Cern, see http://public.web.cern.ch/public/en/spotlight/SpotlightGridFactsAndFiction-en.html), the tier 1 contains the national centres, tier 2 represents the regional centres, tier 3 represents the work groups and finally the tier 4 represents the desktops (Figure 1a). Many works have exploited this topology when proposing replication strategies (Perez et al., 2010; Ranganathan and Foster, 2001; Tang et al., 2005; Liu and Wu, 2006; Shorfuzzaman et al., 2010; Horri et al., 2008). The grid hierarchy usually reflects the structure of organisations in which a potentially large number of replicas are placed at different levels. This explains that requests travel up towards the root. This topology has several advantages. It is easier to implement because of its simplicity. Also, it allows nodes to access the resources in a common and efficient way. Furthermore, the multi-tier structure enables the flexible and scalable management for data sets and users. However, the problem of this topology is the strict rules of a tree structure, i.e. there is only one path available from a leaf to the root, i.e. child (leaf) nodes can communicate only with their direct parent. In consequence, this type of topology is efficient only for data grids which are designed from scratch. Hence, it fails to represent the grid if nodes are randomly added to the system.

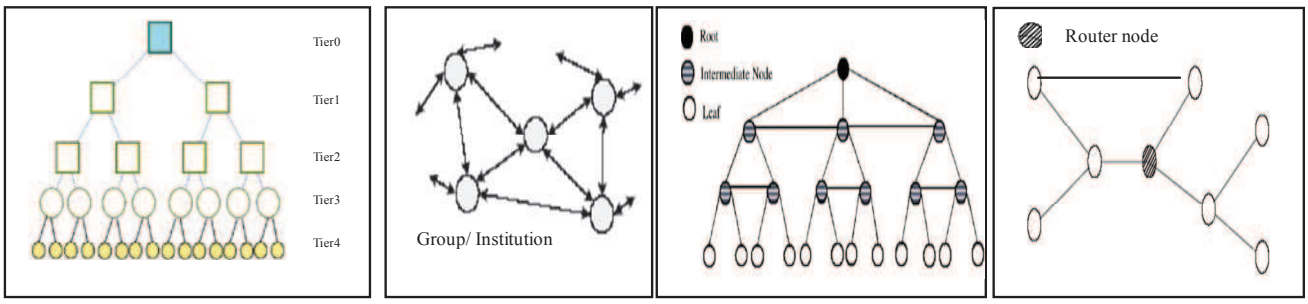
3.2 Federation topology

Most of papers in the literature refer to this topology by the Peer-to-Peer (P2P) topology, which is also called single-tier topology (Figure 1b). One example of a federated data grid is the Bio Informatics Research Network (BIRN). Per opposition to multi-tier topology, all peers in the P2P topology operate independently within a peer group and agree upon a common set of services. It can be represented as a ring between the root nodes of multiple hierarchal structures. On the other hand, a peer can be a member of more than one group at a time and can join or leave a group at any time. P2P systems overcome limitations of the tree structure and offer flexibility in communication among components. Many replication strategies have been proposed under this topology (Ranganathan et al., 2002; Abdullah et al., 2008; Xhafa et al., 2012b) in order to permit a high availability and reliability of data while any replica can synchronise with any other. However, the maintenance of such system generates an important cost.

3.3 Hybrid topology

The hybrid topology integrates the characteristics of hierarchical and federation topologies to get the benefits of both of them. A hierarchical topology is also adopted but nodes at the same level of a tree are connected to each other as shown in Figure 1c. Then, data access among the same tier nodes is allowed. This type of topology, also called sibling tree topology, improves both the data availability and the reliability of the P2P topology and allows for a scalable expansion of the hierarchical topology. Many replication strategies have been proposed under this topology (Rasool et al., 2009). A set of replica management services was proposed by Lamehamedi et al. (2003); while a balanced workload-based replicas placement was proposed by Lin et al. (2008).

Figure 1 Data grid topologies: (a) hierarchical, (b) federation, (c) hybrid and (d) graph topologies (see online version for colours)



(a) (b) (c) (d)

3.4 Arbitrary graph topology

A general arbitrary graph topology (Figure 1d) is a realistic grid topology alternative in which any node can be connected to any other node without any restrictions, i.e. there is no central node designated as a root node. In consequence, developing replication strategies in such topology requires complex protocol since any replica can synchronise with any number of replicas. This explains the fact that only few replication strategy works consider a general graph as a grid topology (Rahman et al., 2008; Lei et al., 2008; Sashi and Thanamani, 2010; Bsoul et al., 2010; Devakirubai and Kannammal, 2013).

3.5 Synthesis

When comparing these topologies, some of them provide more flexibility than others. In a tree topology, every node only accesses the replicas that are in a list of its parent and child locations. In this topology, two types of placing replicas are possible: (a) the first permits to a request to go up and down the tree in order to search the nearest replica (Kalpakis et al., 2001); and (b) the second permits to a request to search a replica towards the root of the tree (Jia et al., 2003). Hence, a replica placement service uses the data grid topology to overlay replicas on the data grid which improves the data access. However, the location of a replica should be carefully considered when carrying out the dynamic replication. In the P2P topology, each replica keeps a list of the locations of its neighbours. Then, peers can decide independently to produce replicas. Although that there is no single point of failure, a decentralised decision may lead to replicas creation of the same file, e.g. a peer may have a partial vision. This motivates the introduction of the hybrid model. In such model, the multi-tier topology increases the availability of data and the P2P topology improves scalability. It has been observed that there exists no standard architecture for a data grid environment. Although most of the work done follows a hierarchal architecture, they have mentioned extending their work to general graphs in the future. The reason is that a general graph is more close to a real-grid environment. In this context, results (Bsoul et al., 2010; Lei et al., 2008; Sashi and Thanamani, 2010), dealing with the general graph topology, are very promising since they were based on real-grid scenarios. However, the frequently arrival/departure of nodes to the system (dynamic property of data grid systems) influences the replication performance in such topology. In consequence, some works, such as Rahman et al. (2008), proposed a replica maintenance algorithm to relocate replicas to other nodes when a candidate node for holding replicas leaves the system or when performance metrics are degraded.

4 Objective function-based replication strategies

Determining data which are the object of the replication should be based on the objective function of the replication strategy. In this section, we propose a replication strategy classification based on objective functions of these strategies.

An objective function is a general method for evaluating the system performance (Sivasubramanian et al., 2004). It serves as a criterion to optimise a replication strategy. Possible objective functions discussed here are: (a) exploit different forms of data locality by considering the popularity of data (Ranganathan and Foster, 2001), (b) advantage the network level locality (Park et al., 2004; Sashi and Thanamani, 2011), (c) maximise economic benefits (Bell et al., 2003a) and (d) exploit a cost model in order to decide a replication while minimising the replication cost (Lamehamedi et al., 2003). Although many strategies are based simultaneously on several objective functions with different levels of inclusion, an objective function can be favoured from the others in a given replication strategy.

A pioneering work was presented by Ranganathan and Foster (2001), in which five (5) distinct replication strategies have been proposed for multi-tier data grid. This work has also included a comparison between these strategies in the perspective of performance. Owing to the success of this work in the literature, most of the proposed replication strategies in the literature have compared their results to those of the five strategies proposed by Ranganathan and Foster (2001). This explains the several proposed replication strategies cited in the next sub-section and based on the first objective function, i.e. data locality.

4.1 Replication strategies based on data locality

Replication strategies based on data locality are adapted to the user queries. They aim to maximise the data locality by exploiting the popularity of data. Popularity of data for replication was initially proposed by Gwertzman and Seltzer (1995). It constitutes an important parameter that most of replication strategies consider by replicating the most requested data. It can be expressed by the number of requests for this data, which is computed by data access rate. In order to handle the fluctuation of data access rate, some works (Lee et al., 2011) apply a periodical collection of data access to determine its popularity, while other works (Lei and Vrbsky, 2006) propose the using of access histories of data to quickly calculate their popularity.

Works of Ranganathan and Foster (2001) are among the first to exploit the popularity of data while replicating data. In addition to the *No replication* strategy, used for comparison with other strategies, Ranganathan and Foster proposed five (5) replication strategies for multi-tier data grid systems:

1 *Plain caching*: a replication is performed every request without any condition,

2 *Best client:* it constitutes the most famous replication strategy dealing with the popularity of data. Each node records the requests history for its file. If the number of requests for each file exceeds some threshold, a replica is created in the node which has the largest number of requests for this file. This node corresponds to the best client for that file,

3 *Cascading*: if the number of accesses for a file exceeds a threshold, a replica is created at the next level on the path to the best client. This process continues through lower levels until it reaches to the best client,

4 *Caching and cascading*: caching and cascading are combined. The requested data is replicated locally in the client node. The client caches data locally, and the server periodically identifies the popular data and propagates them down the hierarchy and,

5 *Fast spread*: the requested file is replicated at each node on the path from the source to the best client. When a client requests a file, a copy is stored at each tier along the path. If the storage on any node is not enough, it removes some file(s) to make room for new replicas. This leads to a faster spread of data.

All these strategies aim to reduce both bandwidth consumption and access latency. For this purpose, they introduced three different types of locality (Ranganathan and Foster, 2001), namely:

1 Temporal locality in which file accessed recently is much possible to be requested again shortly.

2 Geographical locality in which file accessed recently by a client is probably to be requested by adjacent client (the grid hierarchal model usually reflects the geographical locality).

3 Spatial locality in which the related files to recently accessed file are likely to be requested *in the near future*.

By applying different types of locality, replication strategies are different in terms of when, where and how replicas are created or deleted. Ranganathan and Foster (2001) compared performance of the five strategies cited above in the perspective of performance. Three different access patterns were considered: (a) random access, (b) access with temporal locality and (c) access with temporal and small geographical locality. The results indicate that different access patterns need different replication strategies. They also conclude that a significant bandwidth consumption reduction is obtained if the access patterns contain a moderate amount of geographical locality. We describe these access patterns and analyse the comparison results of the five strategies under these access patterns in Section 5.

Although many replication strategies compared their results to these strategies, there are some drawbacks. The best client may not always be the best client, i.e. the client that accesses a file for most of the time may not always keep on accessing the same file. Also, these strategies are simulated under ideal circumstances. For example, algorithms of Ranganathan and Foster (2001) are based on the assumption that the total system replica storage is large enough to hold all the data replica copies. In the next sub-sections, we describe how the most important replication strategies exploit different types of data locality.

4.1.1 Replication strategies based on temporal and geographical locality

Tang et al. (2005) proposed Simple Bottom-Up (SBU) and Aggregate Bottom-Up (ABU) algorithms for multi-tier data grid architecture. The general idea is to exploit the geographical and temporal locality by placing replicas as close as possible to the client on the basis of their popularity. SBU algorithm replicates the file that exceeds a predefined threshold. However, SBU does not well consider the relationship between historical access records. In order to address this problem, ABU is designed to aggregate the historical records to the upper tier until it reaches the root. Simulation results show that the using of ABU decrease both average response time and average bandwidth cost comparing to SBU and fast spread solutions especially when the available storage size of the servers is very small. Wu et al. (2008) interested in how to ensure a load balance among replicas by proposing a placement algorithm that finds the optimal locations for replicas. Authors consider the issue of a geographical locality. Hence, a user may specify the minimum distance it can allow from the nearest data server in order to create the replica. Rasool et al. (2009) proposed a Two-Way Replication (TWR) strategy for hybrid architectures. The most popular data is identified and placed to its proper host in a bottom-up manner. In this way, they are closer to the clients. In the top-down manner, the less popular files are identified and are placed to one tier below the root node. In this way, they are close to the root. Shorfuzzaman et al. (2010) proposed a dynamically create replica for popular data in hierarchal data grid systems. The assumption that popular files have more chances of access in the future is adopted. The proposed Popularity-Based Replica Placement (PBRP) algorithm exploits the geographical locality by placing replicas as close as possible to clients in order to decrease the data access time. Bsoul et al. (2010) proposed the Enhance Fast Spread (EFS) replication strategy for general graph grid architecture. It considers the number and frequency of requests, size of replica and last time the replica was requested while making the replication decision. The simulation results show that EFS performs better than the original fast spread. Nukarapu et al. (2011) proposed a data replication strategy that has a provable theoretical performance guarantee. The key point of this strategy is that when several replicas are available, each node keeps track of the geographical closest replica. The simulation results show that this strategy significantly outperforms popular existing replication strategy under various network parameters. Finally, the Dynamic Hierarchical Replication (DHR) (Mansouri and Dastghaibyfard, 2012) is also based on the geographical locality. Replicas are stored in suitable nodes instead of storing them in many nodes while taking into account of workload capacity of each node.

4.1.2 Spatial locality

Most of the works mentioned above are concentrated on temporal and geographical locality. They have neglected the

spatial locality. This is explained by the fact that replication is usually done after the arrival of the requests which cause a significant delay. In order to reduce this delay, the replication must be done in advance. In this context, Chang et al. (2006) addressed some problems of replication strategies based on temporal locality and also focused on data movement problems by predicting future file needs, i.e. spatial locality. Through a predictive method, the job execution time is reduced by prefetching files which are likely to be requested in the future. As Madi and Hassan (2008) claimed that the growth or decay of accesses is more important factor than access number when determining the popularity of files. In fact, suppose that a file accessed a lot of times in the past and after that, it will not be frequently accessed after a time *t* and the replica will still be created although its popularity is based on the past access number. Lei and Vrbsky (2006) also addressed this problem by proposing the Last Access Largest Weight (LAWL) algorithm for multi-tier data grid systems. In addition to the temporal locality when determining the popularity of files, different weights are given to files according their ages which increase the importance of newer files. In consequence, it gives a more precise metric to determine a popular file for replication and the number of replicas. However, the replica placement is done only in the cluster level and not in a node level. Furthermore, some research works (Khanli et al., 2011) classified it as a centralised method because of the presence of a cluster header which gets file access information from all other headers. Khanli et al. (2011) extended the fast spread strategy, which was proposed by Ranganathan and Foster (2001), by proposing Predictive Hierarchal Fast Spread (PHFS) method designed to decrease the latency of data access in hierarchal data grid systems. It uses predictive techniques to predict the future usage of files. Then, it pre-replicates them on a path from source to client, i.e. the user who works in the same context may be request files with high probability in future.

4.2 Replication strategies based on network level locality

Most of the existing replication strategies try to maximise the data locality in order to reduce data access time. However, the storage capability of each node can be limited. Only small part of data may be supported by data grid nodes since very large amount of data can be produced by data grid. In consequence, effect of data locality is reduced. Some research works take benefit from other form of locality, called 'network level locality'. This type of locality indicates that the requested file is located at the node which has the broadest bandwidth to the node of the job execution. In consequence, the network congestion is one of the objective functions to be optimised. In this context, Park et al. (2004) proposed a dynamic Bandwidth Hierarchy-based Replication (BHR) strategy which benefits from network level locality to reduce data access time by avoiding network congestion in data grids. They divided the nodes into several regions. Then, network bandwidth between nodes within a region will be broader than between nodes across regions. Since bandwidth within region would be larger, BHR tries to maximise the number of required data in the same region in order to fetch replica faster. In this context, a regional popularity of files is considered. However, the BHR strategy has good performance only when the capacity of storage element is small. Other research works (Horri et al., 2008) used the BHR algorithm to address both scheduling and replication problems. Authors affirm that the replica decision is made for long-term optimisation by adopting this strategy. However, the proposed algorithm produces good results especially when the bandwidth hierarchy is clear. Later, Sashi and Thanamani (2011) proposed a modified BHR algorithm to overcome the limitations of the standard BHR algorithm. It increases the data availability by replicating a file in the node where the file has been accessed frequently. This permits us to consider popularity of data in the regional level. Hence, unnecessary replication is avoided and the network is used more effectively. In consequence, less time will be consumed in fetching the required file if this later is presented in a local region. However, searching the best node from all nodes constitutes the main weakness of modified BHR algorithm. Also, data may not be always present in the nearby locations with high bandwidth in data grid environment.

4.3 Replication strategies based on economic behaviours

Economic-based replication strategies try to improve performance through exploiting the dynamism of marketplace and their behaviours. The economic-based replica management strategy was introduced by Sidell et al. (1996) in the Mariposa system. It uses evaluation functions, then decides whether to create local replica or not. It is based on the using of the socio-economic concept 'auction' to select the best replica for a job by using files access patterns. A Storage Broker (SB) participates in these auctions by offering a price at which it will sell access to a replica if it is present. Otherwise, it starts an auction to replicate the requested file onto its storage if it determines that this is economically feasible. Replication strategy of Carman et al. (2002) is also based on the same principle. Each node tries to buy a data item to create the replica at its own node. The value of a file is calculated as the sum of the future payment that will be received by a node. This permits to generate revenue in future by selling them to other nodes. Authors focused on replication optimisation in order to reduce job turnaround time in the long term. They show a significant improvement compared to traditional replication strategies. Research work of Bell et al. (2003a) is similar to Carman et al.'s (2002) with the difference of predicting the costs and benefits through a reverse Vickrey auction protocol. Cameron et al. (2004) applied an auction protocol to select one replica among many. It associates a value with each file using a prediction function. The auction protocol replicates the file only if the potential replica has a higher

value than the lowest valued file currently. Lin et al. (2006b) also proposed a replication strategy based on economic behaviours. A replication broker is used to reduce overheads of replication mechanisms in order to take into account policies regarding data transfer. Later, Abdullah et al. (2012) extended the reverse Vickrey auction protocol that the optimisation agents use for dynamically selecting the best replica of a requested file. Agents used a prediction function for making replication decisions through historical of file access patters. It considers both data locality and network latencies. Finally, Andronikou et al. (2012) presented a QoS-aware data replication mechanism strategy for a system with the centralised architecture. It determines the number of replicas required while considering the infrastructural constraints like the workload balancing on all nodes, bandwidth and the importance of data as well. This later is directly connected to the maximisation of the replication profit, i.e. reputation. This is done by the reduction of the set of data replicas.

4.4 Replication strategies based on cost models

In addition to the estimation of data access gains, replication strategies based on cost models deal with the estimation of both replica creation and replica maintenance costs while their calculation is also based on network latency, bandwidth, replica size. Ranganathan et al. (2002) proposed a replication strategy for P2P topology-based data grid systems. Each peer is independent to take a replication decision whenever data availability is improved. The peer that maximises the difference between the total cost and future benefit of replication implementation is the best client. The advantage of this strategy is that there is no single point of failure when the limit resides in the fact that authors assumed an unlimited amount of storage which is no realistic. Furthermore, this strategy does not consider the network status and requires a minimum number of replicas. Deelman et al. (2002) proposed a replication algorithm based on a cost model for hierarchical tree data grid systems. It uses a cost model to predict whether replicas are worth creating. Simulation results found that it is preferable that leaf client nodes run jobs and higher nodes contained all the storage resources. Although this strategy is very promising, the problem consists in the fact that the results are compared only to the case when no replication was performed. Using of a cost estimation model in replication strategies was also well exploited by Lamehamedi et al. (2003) for hybrid data grid topology. In order to decide whether replication must be performed or not, the improvement in data access gained by replication (benefit) is compared to the cost of a replica creation and its maintenance at run time. A cost function is used to rank the files in the local storage. Then, a replica manager replicates a new file only if it improves the data transfer cost. Parameters which are considered before creating and placing a replica are the access patterns, the storage available at a given node and the cited above estimated costs. Experiments show that the performance gains increase with size of data. Significant improvement in response time is observed and both data transfer costs and bandwidth consumption are reduced. Later, Zhang et al. (2010) construct a probabilistic model for the hierarchical data grid to predict its optimal performance. It shows that the proposed Optimal Replication Algorithm (ORA) is better than three compared replication strategies (ABU, SBU and fast spread).

4.5 Synthesis

Table 1 describes some features of most important dynamic replication strategies we have cited in this paper. Throughout this section, we try to compare the concerned strategies regarding some important characteristics.

The grid topology, for which each strategy is developed, is important and makes strategies different from each other. Replication strategies in the literature have been proposed for four (4) above mentioned grid topologies. However, most of these strategies were developed for the hierarchical grid topology. Although this topology has the advantage of be easy to develop, it imposes some constraints. Only few works deal with a graph topology although this later is the most realistic topology. In this context, interesting results were observed (Bsoul et al., 2010). Dynamic strategies may be implemented either in a centralised or decentralised manner. Advantages and disadvantages of them are given in the related work section. We are based on the results of each strategy to affirm their scalability (Bell et al., 2003a) or no (Lee et al., 2011). It depends upon the topology in which the system is based. Also, some replication strategies, (e.g. Abdullah et al., 2012) do not consider the replication cost consideration. It is also the case for the bandwidth consumption (Rasool et al., 2009). This is done at the cost of improved availability which is considered as the most important objective of almost all replication strategies. On the other hand, most of data replication strategies are validated by simulation. The most commonly simulator used is OptorSim (Bell et al., 2003b). Furthermore, validation of most of them is done through comparison with results of basic strategies such as the Least Recently Used (LRU), the Least Frequently Used (LFU) (Rodriguez et al., 2001), and the fast spread algorithms. The reason lies in the fact that these algorithms are already implemented in OptorSim. However, some replication strategies used their own simulator such as DRepSim (Tang et al., 2005). In few other works, theoretical validations are done through a mathematical and probabilistic modelling of the problem (Zhang et al., 2010). We have also chosen to consider the storage space assumption considered by each strategy. We observe that earlier strategies have considered an unlimited storage capacity which is no realistic (Ranganathan et al., 2002; Carman et al., 2002). However, some recent strategies (Mansouri and Dastghaibyfard, 2012; Andronikou et al., 2012) claim that it is not suitable to make the assumption that many replicas are created as required. For this aim, many algorithms have been proposed to find the optimal number of replicas, which ensures an optimal use of the storage space (Sashi and Thanamani, 2011).

Table 1 Features and classification of some replication strategies

	Objective Function-based classification									
	Centralised vs. Decentralised	*Grid topology*	*Simulation*	*Storage Assumption*	*Scalability*	*Cost Consideration*	*Data locality based*	*Network level locality*	*Economic behaviour*	*Model cost based*
Casanova et al. (2000)	X	Hierarch.	Yes	Unlimited	No	No	–	–	X	–
Ranganathan and Foster (2001)	X	Hierarch.	Yes	Unlimited	No	No	X	–		–
Ranganathan et al. (2002)	X	P2P	Yes	Unlimited	No	Yes	X	–		–
Carman et al. (2002)	X	Hierarch.	Yes	Unlimited	–	Yes	–	–	X	–
Bell et al. (2003)	X	Hierarch.	Yes	Unlimited	Yes	Yes	–	–	X	–
Lamehamedi et al. (2003)	X	Hybrid	Yes	Limited	Yes	Yes	–	–	–	X
Park et al. (2004)	X	Hybrid	Yes	Limited	No	Yes	–	X	–	–
Cameron et al. (2004)	X	Hierarch.	Yes	Limited	–	Yes	–	–	X	–
Tang et al. (2005)	X	Hierarch.	Yes	Limited	No	Yes	X	–	–	–
Lin et al. (2006)	X	Hybrid	No	Limited	No	Yes	X	–	–	–
Chang et al. (2006)	X	Hierarch.	No	Limited	No	Yes	X	–	–	–
Rahman et al. (2008)	X	Graph	Yes	Unlimited	Yes	No	–	–	–	X
Wu et al. (2008)	X	Hierarch.	Yes	Unlimited	No	Yes	X	–	–	–
Rasool et al. (2009)	X	Hybrid	Yes	Limited	No	No	X	–	–	–
Shorfuzzaman et al. (2010)	X	Hierarch.	Yes	Limited	No	Yes	X	–	–	–
Zhang et al. (2010)	X	Hierarch.	No	Unlimited	–	Yes	–	–	–	X
Bsoul et al. (2010)	X	Graph	Yes	Limited	No	Yes	X	–	–	–
Sashi and Thanamani (2011)	X	Hierarch.	Yes	Limited	No	Yes	–	X	–	–
Nukarapu et al. (2011)	X	Hierarch.	Yes	Limited	–	Yes	X	–	–	–
Khanli et al. (2011)	X	Hierarch.	Yes	Limited	No	No	X	–	–	–
Lee et al. (2011)	X	Hierarch.	Yes	Unlimited	No	Yes	X	–	–	–
Abdullah et al. (2012)	X	P2P	Yes	Limited	No	No	X	–	X	–
Andronikou et al. (2012)	X	Hierarch.	Yes	Limited	Yes	Yes	X	–	X	–
Mansouri and Dastghaibyfard (2012)	X	Hierarch.	Yes	Limited	Yes	Yes	–	–	–	X

Regarding the objective function-based strategy classification, most of the strategies cited in Table 1 are based on data locality especially from 2005. This is explained by the extension of the pioneering work of Ranganathan and Foster (2001) in which most of the later proposed replication strategies compared their results. Earlier strategies consider the user queries through only the temporal and geographical locality. Later, some strategies include spatial locality by predicting future data needs which justify a replication in advance. Also, most of works, as shown in Table 1, ignore network latencies. However, this constitutes an important parameter since data may not be always present in the nearby locations with high bandwidth in data grid environment. In this context, some network level locality-based strategies were proposed (Park et al., 2004). In fact, a data transfer save can be possible by placing replicas at nodes with good bandwidth between it and nodes where queries are executed, i.e. this avoid network congestions in a data grid network. Results of replication strategies based on cost models are also very promising. They evaluated creation and maintenance costs of replicas before any replication decision. However, only few strategies have been proposed in this context (Lamehamedi et al., 2003; Rahman et al., 2008). When analysing all these strategies, it is clear that a given replication strategy may favour one objective function over the other. However, once grouped

together, these objective functions better address the issues of replication strategies. This is the case of some recent replication strategies (Abdullah et al., 2012; Andronikou et al., 2012) which favour both the data locality and the economic behaviours.

5 Factors for high performance of data replication strategies

In order to achieve performance while dealing with a data replication process, we need to always ensure that the benefit of a given strategy is higher than the cost of replication (Van Steen and Pierre, 2010). Adopting one strategy rather than another depends of several factors to favourite in order to obtain optimal performance. Hence, trade-offs exist between factors as the access latency, the network state (e.g. bandwidth) and the storage cost in nodes. In consequence, the cost of each replication strategy depends on the decision to favour one factor over others. In what follows, we enumerate the most important factors that impact on performance of any replication strategy.

Optimal granularity: Determining the appropriate granularity of the data to be replicated turns out to be crucial when replicate data with objective of performance. Van Steen and Pierre (2010) demonstrate that optimal granularity depends on applications. For nodes storing static web pages, for example, supporting a replication strategy on a per page basis leads to higher scalability and better performance. Van Steen and Pierre (2010) conclude that replicating for performance requires differentiating replication strategies for smaller data units.

Access latency: Reducing the access latency constitutes an important factor for reducing the job execution time. This is obtaining by sharing information between all nodes in order to find out which data need to be replicated and where to place the new replica (Lei and Vrbsky, 2006). Mansouri and Dastghaibyfard (2012) also reduce access latency by selecting the best replica when multiple nodes hold replicas. The proposed algorithm is based on the response time that can be determined by considering the data transfer time, the storage access latency and the distance between nodes.

Bandwidth consumption: Some replication strategies do not consider an optimal bandwidth consumption since the principal aim is to improve the data availability. However, this factor is very important to ensure performance. In fact, a frequent transfer of data can lead to strain on the network's resource which can impact on performance of the system. This motivates the proposition of replication strategies based on network level we have cited above.

Balanced workload: Placing replicas in optimal locations helps to optimise the workload of the system and then minimise the job execution time. Rahman et al. (2008) proposed a *p-median*-based dynamic replication which find *p* replica placement that minimise distance between the requesting node and the nodes holding replicas. Lin et al. (2006a) focus on the optimal placement of replicas so that the workload of replicas is balanced for the multi-tier architecture.

Access pattern: To prove the impact of the access pattern on replication strategy performance, Ranganathan and Foster (2001) evaluated the performance of five replication strategies with three different access patterns (*random* access pattern, data access with a small amount of *temporal* locality and data access with small amount of *geographical and temporal* locality). Simulation analysis shows that fast spread algorithm performs the best under a random access when cascading technique works better under geographical and temporal locality. To generate the data access pattern with dynamically changed file access popularity on the system, Tang et al. (2005) and Dogan (2009) randomly generate requests according to uniform, geometric and Zipf distributions. In the uniform distribution, the same number of replicas is created for each object (e.g. file) independently of the request. The geometric distribution is used to model the scenario that some data files are requested more times than others. In Zipf distributions, more replicas are created for data that are frequently queried (the number of replicas is proportional to their popularity). Zipf distribution (Breslau et al., 1999) exists widely in the internet world. It means that user's access to file is coherent to time, which is very popular in the file-sharing application of data grid. Dogan (2009) evaluated the performance of eight (8) dynamic replication strategies under different data grid settings. The simulation results show that the file access pattern has great influence on the real-time grid performance. Fast spread enhanced was the best of the eight algorithms considered.

Storage capacity: Although a storage cost is becoming low lately, replication strategies must assume a fixed amount of storage to ensure realism. Replication performance depends significantly on the size of storage available at different nodes and the bandwidth between these nodes. In consequence, there is a trade-off between storage availability and network bandwidth availability (Amjad et al., 2012). One solution consists of a well-designed replica replacement algorithm (Zhao et al., 2010).

Optimal number of replica: Defining an optimal number of replicas in order to avoid the unnecessary replication is an important parameter when replication strategies achieve performance. In fact, maintaining an increased number of replicas can generate an overhead in the system. Lin et al. (2008) focus on the optimal placement of replicas for the hybrid architecture. It tries to maintain a balanced workload on all nodes by proposing an algorithm to find the optimal number of replicas. Then, another algorithm places replicas in optimal locations if both the number of replicas and the maximum allowed workload for each replica have been determined.

Almost replication strategies in the literature consider that improving availability and reducing job execution time constitute the principal aim of these strategies. Although there is a trade-off between some factors, all these factors should be taken into account simultaneously. Keeping data close to the user, i.e. reducing access cost, should not be done at the expense of network congestion. Also, many works have concluded that a good replication strategy must

be based on an efficient replica placement algorithm with an optimal number of replicas while the choice of nodes holding these replicas should not be done at the expense of the system load. The choice of access pattern constitutes also an important factor that impacts on performance of any replication strategy. Although we experiment with classical replication strategies, we discuss in the performance evaluation section why the choice of the access pattern is important when the number of jobs is varied.

6 Cost analysis of data replication strategies

In this section, we analyse the cost of given replication strategy. Pierre and Van Steen (2001) establish a general cost function. Authors consider *m* performance metrics for the deployed replication strategy *s*. Then, a cost $c_k(s)$ is associated for each *k*-th metric. They also associate a weight w_k with the costs $c_k(s)$ which is dependent on *s*. The general cost formula is as following:

$$Rep\_\text{cost}(s) = \sum_{k=1\ldots m} w_k c_k(s)$$

Designed to minimise the total costs of replication, this formula permits to measure and compare strategies. However, the decision to assign a weigh for a strategy back to the administrator. Concerning the metrics used to evaluate performance in a replicas management system, we can classify them into two types: (a) static metrics, e.g. geographical distance, whose estimates do not vary with time per opposition; (b) dynamic metrics, e.g. latency, number of router hops and network usage. In this paper, most of replication strategies we have cited are based on the following metrics:

1 The response time metric is related to the *time* the replication takes for communication between peers. It is generally referred by the latency metrics,

2 The spatial metric is an alternative to temporal metrics. Many systems consider this metric, e.g. number of network level hops. However, Sivasubramanian et al. (2004) demonstrate that although spatial metrics are easier to measure, they are fairly inaccurate as estimators for latency,

3 The bandwidth metric which corresponds to the total amount of consumed network resource,

4 The financial metric is mostly used in the economy-based replication strategies. Some models mandate that the number of replicas of an object is constrained by the money paid by the object owner when other uses peak consumed bandwidth as its pricing metric, and

5 The frequency metric, introduced by Tang et al. (2005). It is defined as how many replications occur per data access. This metric is also important since when there exist many replicas in one server, the workloads of this and its CPU utilisation are affected.

In summary, performance metrics are difficult to compare. For example, one strategy can require low latencies but consumes a lot of bandwidth when another strategy can save network bandwidth at the cost of relatively poor response times. Other works show that there is a trade-off between faster response times and conserving network bandwidth. Ranganathan and Foster (2001) show that if the priority is achieving faster response times, cascading technique might work better but when the priority is to obtain a reduction of bandwidth consumption, fast spread technique is better.

7 Simulation analysis

This section starts with a brief description of the simulation tool we have used. Then, we analyse the obtained simulation results in order to measure the impact of important factors on replication strategy performance.

7.1 Simulation tool and performance environment

In order to measure the impact of some factors on replication strategy performance, we used OptorSim (Bell et al., 2003b), a scalable, configurable and programmable simulation tool. There are three options for replication strategies in OptorSim: (a) no replication which never replicates a file, i.e. data are taken from the master node; (b) LRU and LFU algorithms; and (c) economic model-based replication strategies in which nodes 'buy' and 'sell' files using an auction protocol.

In the LRU algorithm, a requested node always replicates the required data and caches it. Then, if the local storage is full, the oldest replica is deleted to free the storage. However, if the oldest replica size is less than the new replica, the second oldest file is deleted. The LFU strategy performs as the LRU strategy with the difference that it deletes the replica which has a less demand from the local storage even if the replica is newly stored. Concerning the economic-based strategy, there are two types in OptorSim: (a) the binomial economic model based and (b) Zipf economic model based. Several configuration files determine the comportment of OptorSim. In addition of these strategies already implemented in OptorSim for data grid system, we have also simulated the BHR algorithm (Park et al., 2004). Throughout these experiments, we deal with a simulated data grid composed of Computing Element (CE) and Storage Element (SE). Users submit jobs to the system. Then, a Resource Brocker (RB) controls scheduling of jobs to CE nodes according to existing scheduling algorithms (random, shortest queue, access cost, queue access cost). Each node handles its file content with replica manager which, with Replica Optimiser (RO), contains the replication strategies that decide the creation and deletion of replicas. Before starting these experiments, we have initialised several configuration parameters: (a) the general parameters file, e.g. the total numbers of jobs to run, the access pattern choice and the replication strategy are concerned, (b) the grid configuration file, e.g. the network topology, (c) the job configuration file, e.g. the files needed by each job, and (d) the bandwidth configuration file, e.g. the background network traffic. Table 2 describes the principal parameters we have used in this simulation.

Table 2 Configuration parameters

Parameters	*Value*
Number of peers	13
Number of jobs	100
Number of file accessed per job	10
Size of a single data file	1 Gb
Maximum bandwidth between nodes	100 Mb/s

On the other hand, five access patterns exist in OptorSim: (a) sequential, i.e. files are selected at the order stated in the job configuration file, (b) random, i.e. the access follows a random distribution, (c) random walk unitary in which files are accessed using a unitary random walk, (d) random walk Gaussian, i.e. files are requested in a Gaussian distribution and (e) random walk Zipf, in which successive files are selected from a Zipf distribution, i.e. some elements often occur when others occur rarely. Since most of research works show that the distribution of requested web pages generally follows a Zipf distribution (Breslau et al., 1999), we have used the random Zipf access to determine the order in which the files are requested by jobs.

7.2 Simulation results

The first two experiments are based on the following metrics: the mean job execution time and the effective network usage. Then, the mean job execution time is measured when varying the available storage size in a third experiment.

7.2.1 Mean job execution time while varying the grid topology

Throughout these experiments, we have evaluated four replication strategies (LFU, LRU, economic behaviours based and BHR) by varying the grid topology when fixing simultaneously: (a) the storage capacity of all nodes, (b) the number of data files and (c) the bandwidth capacity. We have also varied the grid configuration file. For this aim, we have extended the simple grid configuration already presented in OptorSim2.1 (*simple_grid.conf* file) in order to have the three configurations, i.e. hierarchical, hybrid and graph topologies, as shown in Figure 2. There are four routers (Figure 2a and 2b) and five routers (Figure 2c) that are used to forward requests to other nodes. Jobs are processed in the nodes that have both CE and SE elements. There is a main master node where all data are produced initially. This node has the most important capacity of storage (100 GB) in order to hold all files which are distributed to other nodes (50 GB for each of them).

In the curves of Figure 3, the main execution time corresponds to the total time required to execute all jobs divided by the number of jobs completed (Cameron et al., 2004). We have deliberately chosen to not represent the job execution time when any replication strategy (no replication) is applied. This is because of the problem of scale in these figures. These times correspond to 6105, 6004 and 8200 ms when we experiment with hierarchical, hybrid and graph topologies, respectively, which constitute the most important times when compared to the four algorithms cited above. The BHR algorithm has the shortest mean job execution time in the three curves. This is due to the fact that it locates files and stores them in the most frequently accessed node. When the LRU strategy requires 1545 file accesses to execute all jobs in the hierarchical topology experiment, the BHR algorithm requires only 785 accesses for the same experiment. This is also due to the fact that the minimum distance between the requester node and replicas decreases the job execution time. This explains why a job execution time save is observed with the hybrid topology.

Figure 2 Data grid topology: (a) hierarchical, (b) hybrid and (c) graph topologies

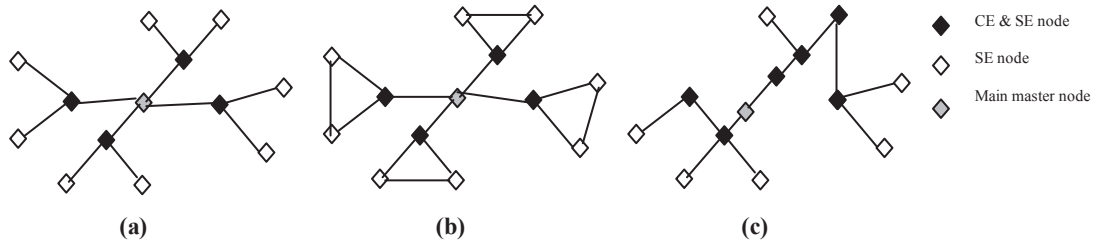
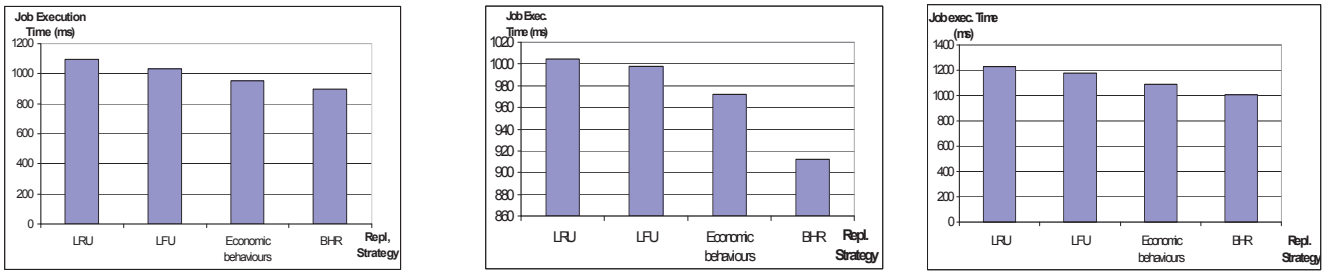


Figure 3 Job execution times for (a) hierarchical, (b) hybrid and (c) graph topologies topologies (see online version for colours)



(a) (b) (c)

Unsurprisingly, the hybrid data grid topology generates the less important job execution times. It profits from the advantages of both hierarchical and P2P topologies. On the other hand, the most important job execution times were observed in experiments when any node is connected to any other without any restrictions of a tree topology. This corresponds to the graph topology in which many synchronisations between nodes are required. Recall curves shown in Figure 3 are obtained when a few files are requested frequently, i.e. Zipf access pattern. When a random access pattern is used, we have observed that LFU and LRU strategies have shorter job execution time.

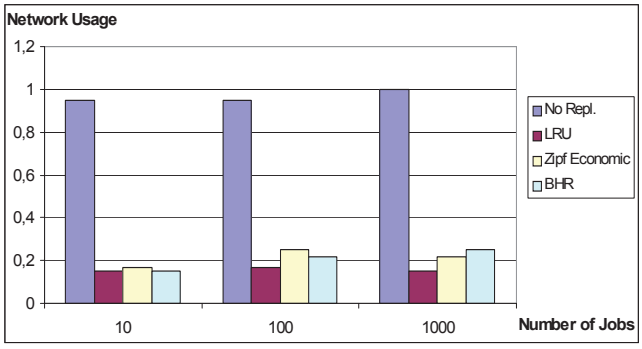
7.2.2 *Effective network usage while varying the number of jobs*

Cameron et al. (2004) define the Effective Network Usage (ENU) as the ratio of files transferred to files requested. Its value ranges from 0 to 1 and corresponds to:

$$ENU = (N_{\text{remote file access}} + N_{\text{file replication}})/N_{\text{local file accesses}}$$

where $N_{\text{remote\_file\_access}}$ is the time required such as a CE reads a file from an SE on a different node, $N_{\text{file\_replication}}$ is the number of replication and $N_{\text{local file accesses}}$ is the time required such as CE reads data from an SE on the same node. Cameron et al. (2004) claim, for a hierarchical topology, that a lower value of ENU indicates that the replication strategy is better. In these experiments, we also deal with the ENU and focus only on the hierarchical topology. In Figure 4, the ENU value is measured for the LRU, Zipf economic and BHR strategies with varying the number of submitted job. We also interest on the case without any replication strategy. While we have *no replication* strategy, the network usage consumption is maximum. The best ENU value is obtained with the LRU strategy. This is due to the fact that replicas are available in all nodes which do not require a network bandwidth to transfer a file from one node to another. BHR strategy presents better results than Zipf economic strategy when experiment with only 10 and 100 jobs. However, Zipf economic profits from the better using of access histories when the number of submitted jobs increased. With the increased number of jobs, using of these access histories decrease the network usage since replication strategies are based on them while deciding to replicate a file.

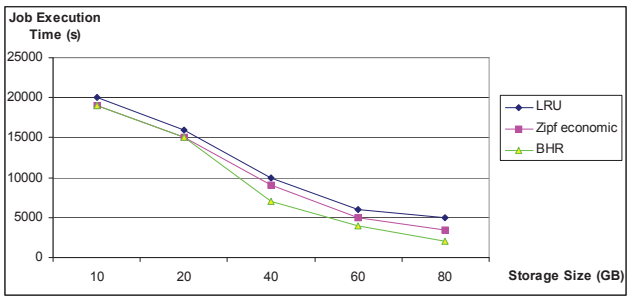
Figure 4 Network usage (see online version for colours)



7.2.3 *Impact of the storage size on the job execution time*

We have also measured the impact of the storage size on the total job execution time (Figure 5). We have observed that when the storage space is enough, all replication strategies have almost the same performance. As the storage size increases, then, the execution time decreases. In fact, the LRU strategy requires the greatest storage size since it replicates always a request that is made when the Zipf economic strategy requires less storage size. However, the BHR method has the lowest storage size requirements since one replica is presented in each region. Then, the storage is done only in some nodes.

Figure 5 Impact of the storage size on execution times topologies (see online version for colours)



8 Related work

Most of related works classified replication strategies according to one of the four following aspects: (a) static vs. dynamic, (b) centralised vs. decentralised, (c) client vs. server initiated replication and (d) unconditional vs. conditional replication.

Regarding the first classification, replica nodes and the number of replicas are chosen statically in advance in the static replication strategy (Chervenak et al., 2002; Tatebe et al., 2002; Bell et al., 2003a). No more replicas are created or migrated after that even if the system changes significantly. Their locations are predetermined, i.e. a replica is persistent until it is deleted by a user or its duration is expired. This type of strategy is suitable when the resource conditions are stable for a long time. Cibej et al. (2005) study the complexity of data replication strategies in data grid systems. They show that this problem is NP-hard. Regarding the advantages of such strategies, they have no overhead of dynamic algorithms and faster job scheduling. However, user behaviour' as access pattern and network condition are varying over time. In consequence, these strategies will not adapt these situations. Thus impacts on the data access efficiency and system performance are affected. Dynamic strategies (Lamehamedi et al., 2003; Chang et al., 2006; Rahman et al., 2008; Mansouri and Dastghaibyfard, 2012) overcome these problems. They allow the system to automatically manage replicas following changing system parameters and its decision depends on different factors as the user access pattern,

storage availability and network bandwidth. In consequence, replicas can be created on new nodes and can be deleted from others. As data grid characteristics are changeable, dynamic replication is more appropriate for data grid. This explains that most of the replication strategies proposed for data grids are dynamic.

Regarding the second classification, a single entity is responsible for deciding on the strategy to adopt, e.g. the placement of replicas, in the centralised process. Several centralised replication strategies have been proposed for different data grid architectures. In the work of Casanova et al. (2000), the popular data are determined by analysing a central data access history in hierarchical data grid systems. This can conduct to a bottleneck especially if there is more than the average load in the network. Ranganathan and Foster (2001) proposed several decentralised replication strategies for hierarchical data grid systems. Another strategy was proposed by Ranganathan et al. (2002) in order to automatically create replicas for a generic decentralised peer-to-peer network, and maintain replica availability with some probabilistic measures. Tang et al. (2005) study the effect of replication schemes and grid-scheduling heuristics on turnaround time. A replication decision is made only at a central dynamic replication scheduler that collects the average number of file accesses in the data access history and clients access pattern. The replication decision (Rasool et al., 2009) is also done in a centralised way for hybrid data grid systems. It is based on a Grid Replication Scheduler (GRS) that consults a certain replica catalogue which administers all the information about the replicas. The most important benefit of a decentralised decision is that there is no single point of failure. Furthermore, a decision does not rely on a central monitoring scheme. The disadvantage is that nodes can make decisions based on partial information, which may lead to unnecessary replication. Other limitation consists in the overhead generated by the invocation of the replica placement service again and again.

Replication strategies can also be classified (Van Steen and Pierre, 2010) into server-initiated vs. Client-initiated replication. Server-initiated replication strategies also called 'push based' (Tang et al., 2005; Rahman et al., 2008; Ranganathan et al., 2002) correspond to most of the replication strategies cited in this paper and are used to enhance performance. The replication decision made by a server can be motivated by observing some factors as access patterns of the user. It can also depend on the number of requests in order to determine the optimal placement to replicate data, e.g. closer to a potential user. In the client-initiated replication also called 'pull based' or 'client-side caching' or caching (Dilley et al., 2002) replicas are created as a result of client requests, independently of any replication strategy, in order to improve access time. Unlike server-initiated replication, which is planned on advance, the caching happens on demand, i.e. caching is a decision made by the client of a resource and not by the owner of a resource. Before passing data to the client, the required data are stored locally in a cache for future use (Nukarapu et al., 2011). Whenever data are requested again, it can be fetched from the cache locally. In fact, caching is viewed as a form of replication (Madi and Hassan, 2008). A good survey on web caching can be found in the work of Rodriguez et al. (2001).

Finally, some works (e.g. Al Mistarihi and Yong, 2008) classified the replication strategies into conditional and unconditional. The unconditional replication consists of performing a replication at every request, i.e. the node that requests a file always stores a copy locally. The plain caching strategy (Ranganathan and Foster, 2001) is an example of unconditional replication strategy. In this context, two algorithms were emerged: (a) the *LRU* and (b) the *LFU* algorithms. However, this type of strategy would make data frequently replicated, which generate unnecessary replication, not suitable in dynamic data grid. Unlike unconditional replication, a conditional replication strategy triggers a creation of replicas according to some conditions such as a popularity threshold. Several replication strategies have been proposed in this context in order to achieve one or several objective functions that we have cited in this paper.

9 Conclusion

Replication strategies have been widely studied in the last decade. The purpose of this paper is to provide a state-of–the-art review concerning the various replication strategies that achieve performance objectives. In consequence, we have focused only on the read-only query scenario. Most related work classified replication strategies into static vs. dynamic or centralised vs. decentralised methods. Other works also classified replication strategies into client vs. server-initiated replication or unconditional vs. conditional replication. In this paper, we propose a new replication strategy classification according to the achieved objective function. We distinguish replication strategies based on: (a) popularity of data while exploiting temporal, geographical and spatial data locality; (b) network level locality; (c) economic behaviours; and (d) cost models. After describing the principal methods for each class, it has been observed that a strategy that promotes only one objective function is not efficient. In consequence, a good replication strategy should include simultaneously several objective functions. On the other hand, although no standard architecture for data grids exists, most replication strategies were developed for the hierarchical data grid topology. However, strict constraints of the tree structure lead us to say that the general graph model is more realistic. In this context, we have cited some replication strategies that provide interesting results. Future proposals should be oriented towards this direction. We are also interested in the different factors that impact on replication strategies performance. We conclude that a good replication strategy must simultaneously consider: (a) the reduction of access time, i.e. promotes data locality; (b) the reduction of the bandwidth consumption; (c) the storage resources availability; (d) a balanced workload between replicas; and (e) a strategic placement algorithm including an optimal number of

replicas. Finding a good balance between them is a good challenge. In order to evaluate their proposed replication strategies, earlier works have based their results in a simulation under the assumed available unlimited amount of storage. However, we believe that a network bandwidth and storage capacity in a data grid may be limited when experiment a new replication strategy. In consequence, the data replication problem is more challenging under the assumption of limited storage resources. In the simulation analysis section, we have measured the impact of some factors that influence performance, e.g. storage availability, and the trade-off between them. We have also measured the impact of the data grid topology on performance of some existing replication strategies. Three different data grid topologies are tested. Best results are obtained with the hybrid data grid topology while the most important job execution times were observed with a graph topology. We also conclude that there are not a lot of comparative studies between replication strategies since each of them, in most cases, promotes the above factors in a separate way. Furthermore, validations of most of these strategies are done through comparison with results of basic strategies such as LFU, LRU and fast spread. The reason lies in the fact that these algorithms are already implemented in existing simulators such as OptorSim. Hence, we believe that comparison with the various other better existing strategies will be required. This can be included in our future work. We also plan to combine replication strategies with scheduling techniques in order to achieve better performance. Another important issue we intend to study is to include the dynamic properties of data grid such that nodes can join or leave the system at each moment. In consequence, replica placement and replica selection algorithms should take into account the dynamic property of data grid environments.

References

Abawajy, J.H. (2004) 'Placement of file replicas in data grid environments', *Proceeding of International Conference (ICCS'04)*, Vol. 3038, pp.66–73.

Abdullah, A., Othman, M., Ibrahim, H., Sulaiman, M.N. and Othman, A.T. (2008) 'Decentralized replication strategies for P2P based scientific data grid', *International Symposium on Information Technology (TSim'08)*, Vol. 3, pp.1–8.

Abdullah, A., Latip, R., Azraei, W.M. and Mustapha, W. (2012) 'Evaluation of an economy-based file replication strategy for Malaysian research and education network (MYREN) data grid model', *Computer and Information Science*, Vol. 5, No. 1.

Al Mistarihi, H.E. and Yong, C. (2008) 'Replica management in data grid', *Proceeding of International Journal of Computer Science and Network Security*, Vol. 8, pp.22–32.

Amjad, T., Sher, M. and Daud, A. (2012) 'A survey of dynamic replication strategies for improving data availability in data grids', *Future Generation Computer Systems*, Vol. 28, pp.337–349.

Andronikou, V., Mamouras, K., Tserpes, K., Kyriazis, D. and Varvarigou, T. (2012) 'Dynamic QoS-aware data replication in grid environments based on data "importance"', *Future Generation Computer Systems*, No. 28, pp.544–553.

Benoit, A. and Rehn-Sonigo, V. (2008) 'Replica placement and access policies in tree networks', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, No. 12, pp.1614–1627.

Bell, W.H., Cameron, D., Carvajal-Schiaffino, R., Millar, A., Stockinger, K. and Zini, F. (2003a) 'Evaluation of an economy-based file replication strategy for a data grid', *Cluster Proceeding of the 3rd IEEE/ACM International Symposium on Computing and the Grid (CCGrid)*, pp.661–668.

Bell, W.H., Cameron, D., Millar, A., Capozza, L., Stockinger, K. and Zini, F. (2003b) 'Optorsim: a grid simulator for studying dynamic data replication strategies', *Proceeding of International Journal of High Performance Computing Applications*, Vol. 17, No. 4.

Bernstein, P.A., Hadzilacos, V. and Goodman, N. (1987) *Concurrency Control and Recovery in Database Systems*, Addison-Wesley Publishers, MA, USA.

Breslau, L., Cao, P., Fan, L., POhillips, G. and Shenker, S. (1999) 'Web caching and Zipf distribution: evidence and implications', *Proceedings of the International Conference on IEEE INFOCOMM*, New York.

Bsoul, M., Al-Khasawneh, A., Eddien Abdallah, E. And Kilani, Y. (2010) 'Enhanced fast spread replication strategy for data grid', *International Journal of Network and Computer Applications*, Vol. 24, No. 2, pp.575–580.

Cameron, D.G., Carvajal-Schiaffino, R., Millar, A.P., Nicholson, C., Stockinger, K. and Zini, F. (2004) 'OptorSim: a grid simulator for replica optimisation', *UK e-Science all Hands Conference*, Vol. 31.

Carman, M., Zini, F., Serafini, L. and Stockinger, K. (2002) 'Toward an economy-based optimization of file access and replication on data grid', *Proceedings of the 2nd International Symposium on Cluster Computing and the Grid*, pp.340.

Casanova, H., Obertelli, G., Berman, F. and Wolski, R. (2000) 'The AppLeS parameter sweep template: user-level middleware for the grid', *Proceeding of the ACM/IEEE Conference in Super Computing*, Denver, pp.60.

Chang, R.S., Huang, N.Y. and Chang, J.S. (2006) 'A predictive algorithm for replication optimization in data grid', *Proceeding of ICS 2006*, Taiyuan, Taiwan, pp.199–204.

Chang, R.S. and Chang, H.P. (2008) 'A dynamic data replication strategy using access weights in data grids', *The Journal of Supercomputing*, Vol. 45, No. 3, pp.277–295.

Chervenak, A., Deelman, E., Foster, I., Guy, L., Hoschek, W., Iamnitchi, A., Kesselman, C., Kunst, P., Ripeanu, M., Schwartzkopf, B, Stockinger, H., Stockinger, K. and Tierney, B. (2002) 'Giggle: a framework for constructing scalable replica location services', *Supercomputing*.

Cibej, U., Slivnik, B. and Robic, B. (2005) 'The complexity of static data replication in data grids', *Parallel Computing*, Vol. 31, pp.8–9, pp.900–912.

Deelman, E., Lamehamedi, H., Szymanski, B. and Zujun, S. (2002) 'Data replication strategies in grid environments', *Proceedings of 5th International Conference on Algorithms and Architecture for Parallel Processing, ICA3PP'2002*, IEEE Computer Science Press, Beijing, China, pp.378–383.

Devakirubai, N. and Kannammal, A. (2013) 'Optimal replica placement in graph based data grids', *The International Journal of Engineering and Science (IJES)*, Vol. 2, No. 3, pp.95–103.

Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R. and Weihl, B. (2002) 'Globally distributed content delivery', *IEEE Internet Computing*, Vol. 6, No. 5, pp.50–58.

Dogan, A. (2009) 'A study on performance of dynamic file replication algorithms for real-time file access in data grids', *Future Generation Computer System*, Vol. 25, pp.829–839.

Garcia-Carballeira, F., Carretero, J., Calderon, A., Garcia, J.D. and Sanchez, L.M. (2007) 'A global and parallel file systems for grids', *Future Generation Computer Systems*, Vol. 23, No. 1, pp.116–122.

Goel, S. and Buyya, R. (2006) 'Data replication strategies in wide area distributed systems', *Enterprise Service Computing: From Concept to Deployment*, pp.211–241.

Gwertzman, J.S. and Seltzer, M. (1995) 'The case of geographical push-caching', *Proceedings of International Workshop on Hot topics in Operating Systems (HotOS-V)*, pp.51–55.

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

Jia, X., Li, D., Hu, X-D., Wu, W. and Du, D-Z. (2003) 'Placement of web-server proxies with consideration of read and update operations on the internet', *Computer Journal*, Vol. 46, No. 4, pp.378–390.

Kalpakis, K., Dasgupta, K. and Wolfson, O. (2001) 'Optimal placement of replicas in trees with read, write, and storage costs', *IEEE Transactions on Parallel Distributed Systems*, Vol. 12, No. 6, pp.628–637.

Khanli, L.M., Isazadeh, A. and Shishavanc, T.N. (2011) 'PHFS: a dynamic replication method, to decrease access latency in multi-tier data grid', *Future Generation Computer Systems*, pp.233–244.

Kunszt, P., Laure, E., Stockinger, H. and Stockinger, K. (2005) 'File-based replica management', *Future Generation Computer Systems*, Vol. 22, No. 1, pp.115–123.

Lamehamedi, H., Shentu, Z., Szymanski, B. and Deelman, E. (2003) 'Simulation of dynamic data replication strategies in data grids',

Lee, M.C., Leu, F.Y. and Chen, Y. (2011) 'PFRF: an adaptive data replication algorithm base on star topology data grids', *Future Generation Computer Systems* (2011).

Lei, M. and Vrbsky, S. (2006) 'A data replication strategy to increase availability in data grids', *Proceedings of International Conference in Grid Computing and Applications*, Las Vegas, NV, pp.221–227.

Lei, M., Vrbsky, S.V. and Hong, X. (2008) 'An on-line replication strategy to increase availability in data grids', *Future Generation Computer Systems*, Vol. 24, No. 2, pp.85–98.

Lin, Y.F., Liu, P. and Wu, J.J. (2006a) 'Optimal placement of replicas with locality assurance', *Proceedings of the International Conference on Parallel and Distributed Computing*.

Lin, Y.F., Wu, J.J. and Liu, P. (2008) 'A list-based strategy for optimal replica placement in data grid systems', *37th International Conference on Parallel Processing*, pp.198–205.

Lin, H., Abawajy, J.H. and Buyya, R. (2006b) 'Economy-based data replication broker', *Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing*, Amsterdam, the Netherlands.

Liu, P. and Wu, J.J. (2006) 'Optimal replica placement strategy for hierarchical data grid systems', *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, pp.420–423.

Loukopoulos, T., Lampsas, P. and Ahmad, I. (2005) 'Continuous replica placement schemes in distributed systems', *Proceedings of 19th International Conference on Supercomputing*, pp.284–292.

Madi, M.K. and Hassan, S. (2008) 'Dynamic replication algorithm in data grid: survey', *Proceedings of International Conference on Network Applications, Protocols and Services*, Malaysia.

Mansouri, N. and Dastghaibyfard, G.H. (2012) 'A dynamic replica management strategy in data grid', *Journal of Network and Computer Applications*, Vol. 35, No. 4, pp.1297–1303.

Nukarapu, D.T., Tang, B., Wang, L. and Lu, S. (2011) 'Data replication in data intensive scientific applications with performance guarantee', *Proceedings of IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, pp.1299–306.

Park, S.M., Kim, J.H., Ko, Y.B. and Yoon, W.S. (2004) 'Dynamic data grid replication strategy based on internet hierarchy', *Grid and Cooperative Computing*, pp.838–846.

Perez, J.M., Garcia-Carballeira, F., Carretero, J., Calderon, A. and Fernandez, J. (2010) 'Branch replication scheme: a new model for data replication in large scale data grids', *Future Generation Computer Systems*, Vol. 26, No. 1, pp.12–20.

Pierre, G. and Van Steen, M. (2001) 'Globule: a platform for self-replicating web documents', *Proceedings of the International Conference on Protocol for Multimedia Systems*, Enschede, the Netherlands, pp.1–11.

Rahman, R.M., Barker, K. and Alhajj, R. (2008) 'Replica placement strategies in data grid', *Journal of Grid Computing*, Vol. 6, No. 1, pp.103–123.

Ranganathan, K. and Foster, I. (2001) 'Identifying dynamic replication strategies for a high performance data grid', *International Workshop on Grid Computing*.

Ranganathan, K., Iamnitchi, A. and Foster, I. (2002) 'Improving data availability through dynamic model-driven replication in large peer-to-peer communities', *Cluster Computing and the Grid (CCGrid)*, pp.376.

Rodriguez, P., Spanner, C. and Biersack, E. (2001) 'Analysis of web caching architecture: hierarchical and distributed caching', *IEEE/ ACM Transaction on Networking*, Vol. 21, 4 (Aug.), pp.404–418.

Rasool, Q., Li, J. and Zhang, S. (2009) 'Replica placement in multi-tier data grid', *8th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp.103–108.

Sashi, K. and Thanamani, A.S. (2010) 'A new dynamic replication algorithm for European data grid', *Proceedings of the 3rd Annual ACM Bangalore Conference*, pp.17.

Sashi, K. and Thanamani, A.S. (2011) 'Dynamic replication in a data grid using a modified BHR region based algorithm', *Future Generation Computer Systems*, Vol. 27, No. 2, pp.202–210.

Shorfuzzaman, M., Graham, P. and Eskicioglu, R. (2010) 'Adaptive popularity driven replica placement in hierarchical data grids', *Journal of Super Computers*, Vol. 51, pp.374–392.

Sidell, J., Aoki, P., Barr, S., Sah, A., Staelin, C., Stonebraker, M. and Yu, A. (1996) 'Data replication in mariposa', *Proceedings of 17th International Conference on Data Engineering*, New Orleans, USA, pp.485–494.

Sivasubramanian, S., Szymaniak, M., Pierre, G. and Van Steen, M. (2004) 'Replication for web hosting systems', *ACM Computer Survey*, Vol. 36, No. 3, pp.1–44.

Tang, M., Lee, B.S., Yeo, C.K. and Tang, X. (2005) 'Dynamic replication algorithms for the multi-tier data grid', *Future Generation Computer Systems*, Vol. 21, No. 5, pp.775–790.

Tatebe, O., Morita, Y., Matsuoka, S., Soda, N. and Sekiguchi, S. (2002) 'Grid data farm architecture for Petascale data intensive computing', *CCGrid*, pp.102.

Tu, M., Li, P., Xiao, L., Yen, I-L. and Bastani, F.B. (2006) 'Replica placement algorithms for mobile transaction systems', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 18, No. 7, pp.954–970.

Van Steen, M. and Pierre, G. (2010) 'Replicating for performance: case studies', in Charron-Bost, B., Pedone, F. and Schiper, A. (Eds): *Replication, Theory and Practice*, Vol. 5959, Springer, Berlin, pp.73–89.

Wu, J.J., Lin, Y.F. and Liu, P. (2008) 'Optimal replica placement in hierarchical data grids with locality assurance', *Journal of Parallel and Distributed Computing*, Vol. 68, No. 12, pp.1517–1538.

Xhafa, F., Potlog, A-D., Spaho, E., Pop, F., Cristea, V. and Barolli, L. (2012a) 'Evaluation of intra-group optimistic data replication in P2P groupware systems', in Fox, G.C. and Moreau, L. (Eds): *Concurrency and Computation: Practice and Experience*, John Wiley & Sons, Ltd.

Xhafa, F., Kolici, V., Potlog, A., Spaho, E., Barolli, L. and Takizawa, M. (2012b) 'Data replication in P2P collaborative systems', *Proceedings of the 7th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, pp.49–57.

Zhang, J., Lee, B.S., Tang, X. and Yeo, C.K. (2010) 'A model to predict the optimal performance of the hierarchical data grid', *Future Generation Computer Systems*, Vol. 26, No. 1, pp.1–11.

Zhao, W., Xu, X., Wang, Z., Zhang, Y. and He, S. (2010) 'A dynamic optimal replication strategy in data grid environment', *Proceedings of International Conference on Internet Technology and Applications*, pp.1–4.