A classification of file placement and replication methods on grids

Jianwei Ma[1], Wanyu Liu[1,2], Tristan Glatard[2]

[1] *HIT-INSA Sino-French Research Center for Biomedical Image, Harbin Institute of Technology, Harbin 150001, CHINA*

[2] *CREATIS ; CNRS ; Université de Lyon ; INSERM ; 69621 Villeurbanne, FRANCE*

Abstract

This paper presents a classification of file placement and replication methods on grids. The study is motivated by file transfer issues encountered in the Virtual Imaging Platform deployed on the European Grid Infrastructure. Approaches proposed in the last 6 years are classified using taxonomies of replication process, replication optimization, file models, resource models and replication validation. Most existing approaches implement file replication as a middleware service, using dynamic strategies. Production approaches are slightly different than works evaluated in simulation or in controlled conditions which (i) mostly assumes simplistic file models (undistinguished read-only files), (ii) rely on elaborated access patterns, (iii) assume clairvoyance of the infrastructure parameters and (iv) study file availability less than other metrics but insist on cost.

Keywords: File placement, replication, taxonomy, classification, grid.

1. Introduction

Data management is a critical component of distributed systems and in particular grids [1, 2, 3]. Usually, applications do not seek only for computing power but also have stringent requirements concerning data sharing, storage and transfers. For instance, medical image analysis often involves pipelines or workflows retrieving images from an indexed database, staging-in image files to the computing resources and storing results. This requires that file storage and transfers are secure, reliable and efficient.

Among data management issues, file placement and replication were studied since the early ages of grid[1]. However, applications running on production grids still report low transfer reliability and performance. In [4], data transfers between grid sites account for 86% of job failures and [5, 6] mention that data-related errors have serious impact on the performance. Such data transfer issues originate in the unavailability of storage machines and in network connectivity issues between sites. They not only impact computing jobs but also end-users in their transfer of input and output data.

[1] `http://eu-datagrid.web.cern.ch/eu-datagrid`

Groups of users, a.k.a. Virtual Organizations (VOs), also have to properly deal with file placement in order (i) to control available space on storage machines and (ii) to reduce the impact of storage decommissioning.

File replication is a common solution to improve the reliability and performance of data transfers. It consists in storing several instances of the same file on different resources. File placement aims at choosing these resources appropriately. For instance, [7] reports significant improvements obtained by replication on metrics such as makespan, job running time and consumed bandwidth. Yet, the overhead of file replication also has to be considered [8].

Many file management strategies were proposed but none was adopted in large-scale production infrastructures. Although the European Grid Infrastructure (EGI) reports average availability and reliability ranging from 84% to 96% with a target of 90% [2] (which clearly motivates the need for efficient data management strategies), deployed middleware systems still offer very low-level data management facilities, putting file placement and replication decisions in the hands of the application or VO managers. For instance, high-energy physics experiments[3] still do not expect the middleware to provide any data placement system due to the complexity and variety of application use-cases [9]. In practice, they rely on substantial manpower to implement data placement policies: system administrators constantly monitor the status of the data sets and storage systems, triggering transfers when needed [10].

Such manual operations are not affordable for smaller user communities. Instead, automatic file placement methods should be made available to enable reliable use of grid infrastructures. This paper aims at providing a structured outline of the existing file placement and replication methods. It extends the taxonomies of replication architecture and strategy presented in [11]. Our study is motivated by the practical example of the Virtual Imaging Platform (VIP) [12], a science-gateway deployed on EGI. On this platform, file transfer performance and reliability are a real issue, and we are looking into strategies to improve them. Section 2 describes file transfer issues VIP. Section 3 then formalizes the problem and Section 4 presents a taxonomy of the approaches. Existing works are classified according to this taxonomy in Section 5, and production systems are discussed in Section 6. This paper focuses on grids as aggregations of computing and storage clusters: desktop grids, volunteer computing, clouds, peer-to-peer systems and parallel file systems used for clusters (NFS[4], Lustre[5], PVFS [14], Hadoop [15], GFS [16]) are out of scope.

2. File transfers in the Virtual Imaging Platform

Applications in VIP are described as workflows generating jobs distributed to EGI sites. When they reach a computing resource, jobs download their input files from storage elements (SEs), run the application, and upload results to storage elements. To

[2] https://documents.egi.eu/document/413

[3] The largest grid user community regarding the volume of processed data.

[4] `http://rsync.tools.ietf.org/html/rfc3530`

[5] `http://www.lustre.org`

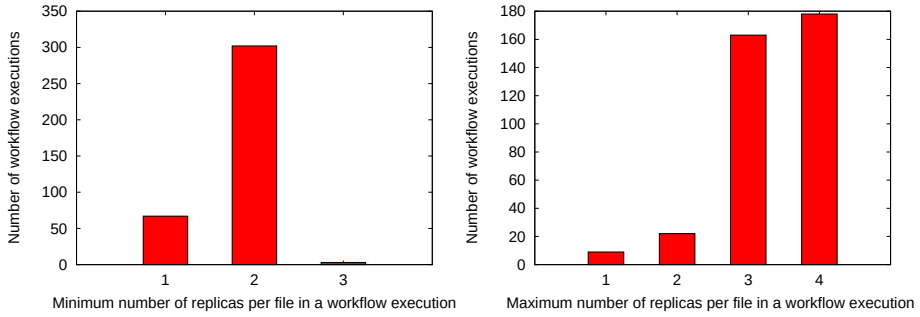

Figure 1: Distributions of minimum (left) and maximum (right) number of replicas per file in a workflow execution

motivate our study of file placement and replication, we studied the performance and reliability of job file transfers in VIP on a data-set of 489 workflow executions started in November 2012 on EGI's `biomed`[6] Virtual Organization.

When workflow input files are uploaded to VIP, they are replicated up to four times to SEs specified in the configuration. Figure 1 shows that most files involved in workflow executions are replicated at least twice. The data-set also shows that in 9% of the workflows, there is no SE holding a replica of all the files, in another 9% of the workflows, 1 SE holds a replica of all the files, and in the remaining 82%, 2 SEs hold a replica of all the files. This is consistent with the fact that SEs in the configuration are used to store the input data. The situation of output files is very different. Output files are uploaded to the SE closest to the computing site, resulting in a wide distribution of files with only 1 physical copy.

To highlight the problems resulting from such a naive file placement strategy, and to motivate our study of replication strategies, we studied the total download time per successful job. After removing workflows with incomplete data, 4639 jobs and 4599 file downloads belonging to 152 workflow executions remained. Figure 2 plots the job download times as a function of the download size. Download sizes are roughly clustered in 6 groups for which whisker bars figure the transfer time quartiles (min, first quartile, median, third quartile, max). For each size group, the inter-quartile range (IQR) of the download time is much larger than the median. This indicates very heterogeneous performance among jobs, resulting in poor quality of service. Figure 3, which plots job download times per grid site for the 6th size group, shows that such heterogeneity is correlated to the geographical locality of the jobs: depending on the site, median transfer times per site range from 20s to 826s. Site 1 stores a copy of most files, which explains that it has the lowest median and IQR. From these observations, we expect that improving file placement on grid sites will largely reduce performance heterogeneity among jobs.

We also studied the job failure rate due to data transfer in workflow executions. Figure 4 plots the cumulative density function of the job error rate in the 410 work-

[6] `http://lsgc.org/en/Biomed:home`

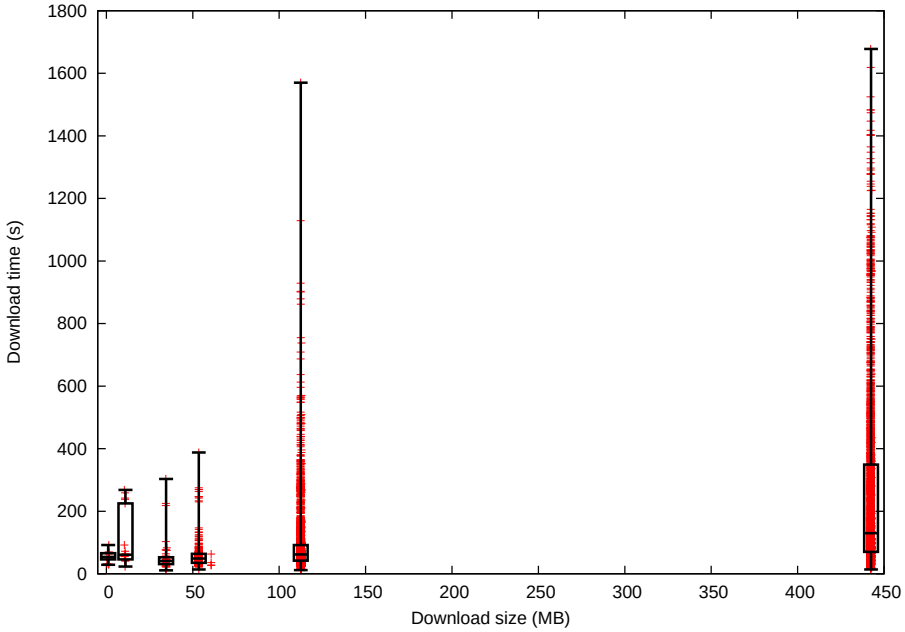

Figure 2: Job download times w.r.t download sizes. For each of the 6 size group, whisker plots represent the transfer time quartiles.

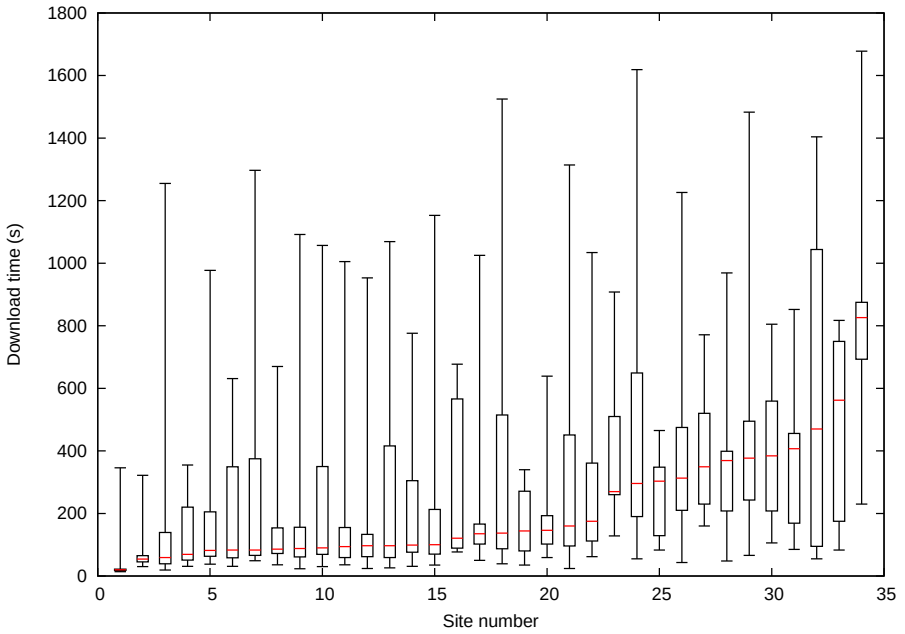

Figure 3: Job download time quartiles per site, for file sizes larger than 400MB (6th size group on Figure 2). Sites are ordered by increasing median values.

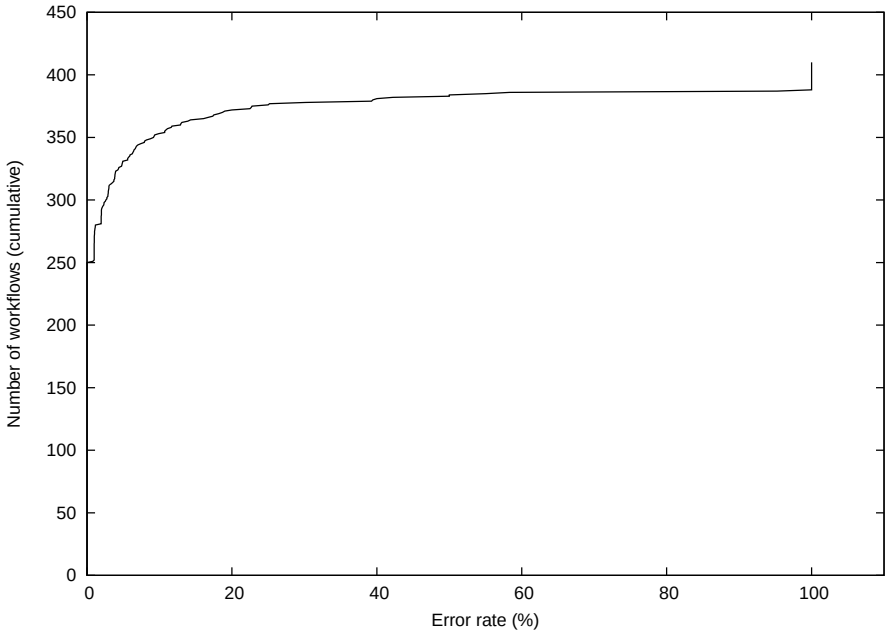

Figure 4: Distribution of job error rate due to failed data transfers in workflow executions.

flows of the data-set with at least 1 job. Only errors coming from file download are considered. Although 60% of the workflow executions are not impacted by download errors (250 out of 410), 20% of the executions have more than 5% of errors (78 out of 410). It means that for 78 workflows out of 410, more than 5% of the jobs failed due to download errors. We expect that such an error rate could be largely reduced by improving VIP's file replication strategy.

3. Problem formalization

We define the file placement problem as follows. Given a set of n files and p storage elements, a storage configuration is defined by a matrix S of $n \times p$ boolean values such that $S_{i,j} = 1$ if and only if file i is stored on storage element j. The set of storage matrices is noted $\mathbb{S}$.

Implementing a storage matrix S means performing all the required file transfers and replications so that the status of the infrastructure is described by S. This has a cost defined by the migration function:

$$\begin{aligned} \phi : \mathbb{S} \times \mathbb{S} &\rightarrow \mathbb{R}^+ \\ S, P &\mapsto \phi(S, P), \end{aligned}$$

where S is the storage matrix before the migration and P is the storage matrix after the migration. Note that ϕ is a distance when file deletion/creation are not considered and

the infrastructure does not change[7].

Each file has a fixed size and storage elements have a capacity. Files have to be stored on at least one storage element but they can be replicated several times: $\sum_{j=1}^{p} S_{i,j} \geq 1$.

Metrics may be associated to a storage matrix. Some metrics may be exactly determined directly from the storage matrix (e.g. total used storage space) and other ones may only be estimated and/or may require additional information about the infrastructure and its usage (e.g. total data transfer time of an application that uses a given storage matrix).

The file placement problem consists in defining a storage function f giving the storage matrix to be realized. The storage function may depend on various parameters characterizing the file placement problem. For instance, it may depend on time (dynamic problem), current value of the storage matrix, file properties (e.g., access patterns, owner, content), infrastructure characteristics and storage cost.

A file placement algorithm implements a storage function optimizing the metrics of interests.

4. Taxonomies of file placement

To create the taxonomies we use the method proposed and used in [17] for a review of multi-criteria grid workflow scheduling. It is described as follows (see beginning of Section 3 in [17]):

> [To analyze the file placement problem], several important *facets* of the problem are considered. Each facet describes the [file placement] problem from a different perspective. For every facet we propose a certain *taxonomy* which classifies different [file placement] approaches into different possible *classes*. Classes are described using the RDF notation *subject-predicate-object*, which are extended in some cases to distinguish between different *sub-classes* of the problem.

Here we analyze the file placement problem in 5 facets that are the file model, the resource model, the optimization criterion, the replication process and the method validation.

The file model includes information about how the applications create and access files. Together with the resource model, it defines the assumptions of the problem. The optimization criterion defines the goal to achieve while the replication process defines the solution. Although the validation method is not really part of the replication method, it gives relevant insight on the scope and applicability of the solution.

Each of these facets is a *subject* of one or several *predicates*: if facet F is a subject of predicate P then it means that F *has a* P. Each predicate has least two *objects* that can be concrete or abstract classes. Concrete classes are not inherited. Depending on

[7] Indeed, $\phi(S,P) = 0 \Leftrightarrow S = P$; $\phi(S,P) = \phi(P,S)$ (if P and S store the same files, have the same storage elements and network links are assumed symmetric) and $\phi(S,Q) + \phi(Q,P) \geq \phi(S,P)$.

the predicate, inheritance can be mutually exclusive or not. Replication methods are characterized by a set of instances of concrete object classes.

4.1. Taxonomy of file models

Figure 5 shows our taxonomy of file models. The tree root represents the facet (ellipse, black text), level-1 nodes are predicates (rectangles, blue text) and level-2 nodes are concrete classes (rectangles, green text). The first predicate is the file type. In some cases a distinction is made between application input and output files. Input files are potentially transferred to several computing sites for processing. They are supposedly replicated on sites that are likely to receive jobs. Output files are application results, used only for user inspection or light post-processing by a reduced number of jobs. A common strategy is to store them close to the site where they are produced. This difference between input and output files can be important *a-priori* information for the replication process.

Similarly, permanent and temporary files may be considered as different types. Permanent files are usually input or output files. They have to be available over a time period spanning several days or weeks and are potentially transferred by the user for visual inspection. Temporary files can be intermediate results. They are not meant to remain available for a long time but they can require substantial amounts of storage and optimization [18].

Instead of individual files, the replication manager sometimes considers *collections* of files belonging to a particular data set (e.g. images of the same patient or events recorded on a particular day). Note that collections do not include groups of files with no semantic relation (e.g. group of files stored on the same host). Considering collections reduces the complexity of the replication problem and ensures that consistent groups of files are reached with similar performance (latency, throughput and availability). Collections, a.k.a groups, blocks or datasets are for example used by high-energy physics experiments.

The file model can also take into account information about file access patterns. Formally, the file access pattern of n files from q nodes during time interval $[t_1, t_2]$ can be described by a matrix A of $n \times q$ integer values such that $A_{i,j}^{[t_1,t_2]}$ is the number of accesses of file i from storage element q in $[t_1, t_2]$. In practice, nodes can be computing nodes or user stations. We also can define $A_i^{[t_1,t_2]} = \sum_{j=1}^{q} A_{i,j}^{[t_1,t_2]}$, the total number of accesses of file i. For readability purposes, time intervals may be omitted.

In some cases assumptions are made on the number of file accesses over a time period. For instance files can be uniformly accessed over a time period, i.e., $A_i = A_j$, $i, j \in [\![1, n]\!]$. A probabilistic distribution of density f can also be assumed: $f(n) = P(A_i = n)$. Usually, f does not depend on the file. Various distributions are used among which uniform, Gaussian [19], binominal and Zipf [20].

In other cases a temporal relation is assumed among file accesses, i.e, an assumption is made on A_i as a function of time ($A_i : t \mapsto A_i^{[t,t+\delta t]}$, $\delta t > 0$). For instance, recently accessed files can be assumed more likely to be accessed again if A_i is an increasing function.

Geographical locality [21] means that the accesses of file k from node i and its neighbors are related. This can be written $|A_{k,i} - A_{k,j}| = g(i, j)$, where g is a function

of the geographical distance between node i and node j. The geographical distance can be defined in several ways, for instance using network hops.

Spatial relations between files can also be assumed, i.e files located close to a recently accessed file are likely to be accessed again. In this case a relation between the storage matrix and the access pattern is assumed: $|A_i - A_j| = h(\{(x, y), S_{i,x} = 1, S_{i,y} = 1\})$, where h is a function depending on the distances between storage elements.

Finally, a workflow relation can also hold, where file dependencies are defined by a graph of data dependencies between jobs. In general, workflows are assumed abstract which means that they only specify dependencies among jobs but contain no information about the execution resources[8]. In this case a function w gives A_i on the time period of the workflow execution: $A_i = w(i)$. This function defines relations between files accesses. For instance, if two files i and j are processed by the same amount of jobs then $w(i) = w(j)$. Temporal dependencies cannot be completely expressed by the workflow since they depend on job scheduling. However, constraints corresponding to dependencies among jobs can be set by w. For instance, if file i is produced by a job and used only by a subsequent job to produce file j then $w(i, t) > w(j, t)$.

File permissions are another important predicate of the file model. Assuming that files cannot be modified simplifies the handling of consistency issues among replicas. In case file deletion is possible the replication process has to consider the file deletion cost that is dependent on the number and placement of the replicas.

Files can originate from a single or from multiple data production sources such as in [21], [22], [19] and [23]. Multiple data production sources cover data produced by analysis jobs or uploaded by geographically distributed users or devices.

4.2. Taxonomy of resource models

The organization and interactions among storage and computing resources define a resource model on which file placement and replication strategies rely. Our taxonomy of resource models is shown on Figure 6. Rectangle with red text indicate abstract classes.

The infrastructure topology is the first important predicate. It can be single-tier or multiple-tier. In a single-tier topology all sites play the same role w.r.t. file placement. Multiple-tier infrastructures give salient roles to some storage elements such as storing output files or holding a copy of all primary files. In case data is produced from a single source, sites can also be ordered w.r.t. their distance to the source.

Different site models can also be envisaged. Sites may consist of storage elements only, of computing resources only or of a combination of both.

Assumptions about clairvoyance with respect to the resource model have a strong impact on the applicability of file placement strategies. Clairvoyant models assume that resource characteristics of interest are entirely known to the file placement algorithm. This knowledge can be either obtained ***a-priori*** or estimated from experimental measures such as done in [8] for network bandwidths. This is often not realistic in large-scale infrastructures where non-clairvoyance may concern the size of available

[8] i.e. a workflow relation only make assumptions on A_i, not $A_{i,j}$

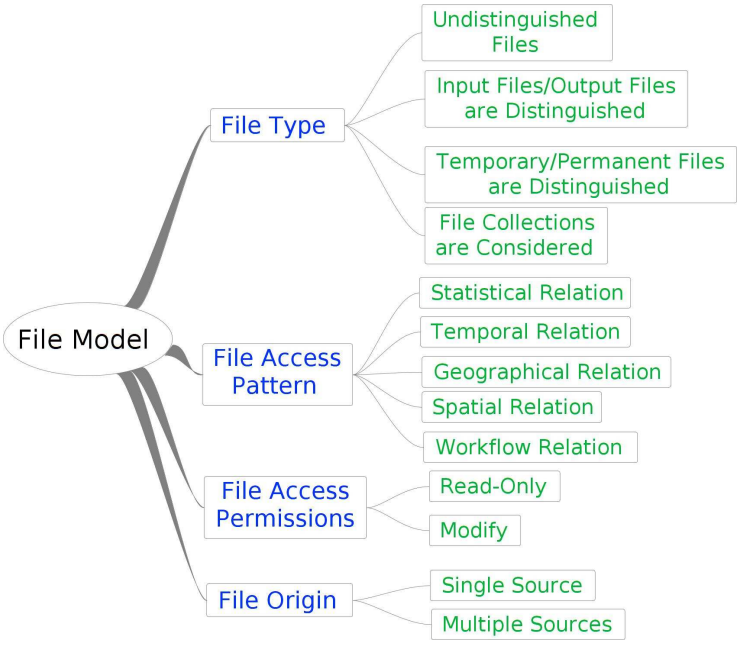

Figure 5: Taxonomy of file models.

.

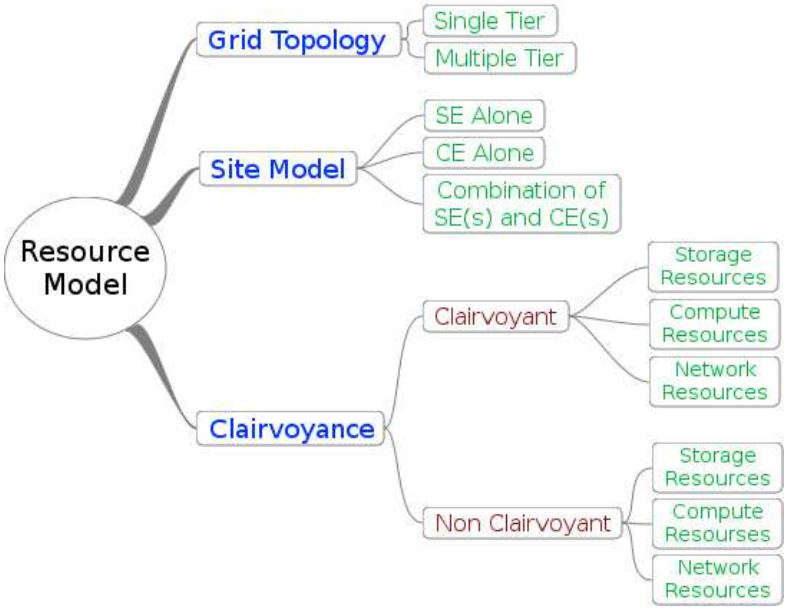

Figure 6: Taxonomy of resource models.

.

storage on SEs, the network throughput and bandwidth between SEs and computing resources, the performance of computing elements or any combination of these.

4.3. Taxonomy of optimization criterions

Our taxonomy of optimization criterions is shown on Figure 7. In case the infrastructure is the optimization target, criteria such as the available network bandwidth or the system file missing rate are used. Instead, works targeting application metrics consider criteria such as the makespan, job running times or throughput of application data transfers.

Besides, replication optimization can be conducted within different scopes. In a local scope the strategy aims at optimizing the placement of only a sub-group of files while in a global scope all the files of the infrastructure are considered. A strategy is said global if and only if there is a defined set of resources that are dedicated to the set of files considered by the strategy. For instance, if a VO has storage quotas on storage elements of the infrastructure, then a strategy optimizing the placement of files of this VO is global, even if other VOs may also have storage quotas allocated on the same storage elements. Local strategies may aim at optimizing file placement for a particular user or application run.

An important characteristic of the file placement optimization is the targeted metric. Metrics can be related to the transfer *time* (e.g. file transfer time, job running time) or to the file availability (e.g. system file/byte missing rate). In some cases metrics depend

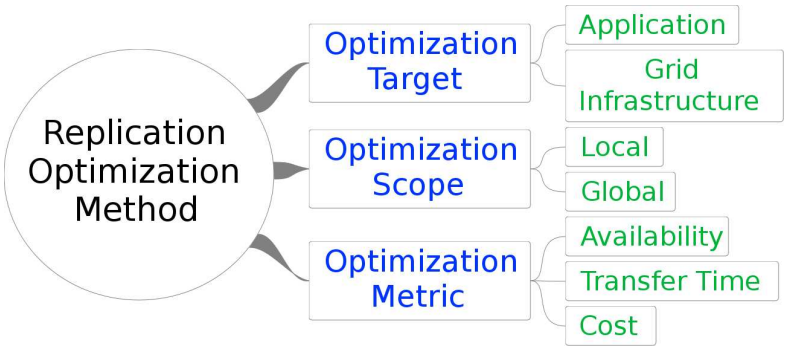

Figure 7: Taxonomy of the replication optimization criterions.

.

both on the availability of files and on their transfer time. This is for instance the case of the makespan of an application when jobs have to be rescheduled when required files are not available. Metrics can also be related to some cost function, for instance network consumption, storage usage, power consumption or even economic cost.

4.4. Taxonomy of replication processes

The taxonomy of replication processes is shown on Figure 8. A replication process can be centralized or decentralized. In case of a centralized process a single entity is responsible for deciding on the placement strategy. For instance this entity can be a job scheduler or a dedicated data management service such as a file catalog. In a decentralized process file placement decisions are taken by multiple entities that may or may not communicate. These can be computing jobs, storage sites or individual users. An example of central replication strategy is described in [23] while [24] presents a decentralized method where replication actors are users.

The replication process is also characterized by its dynamism, i.e., its ability to change the storage matrix along time. A strategy is said dynamic if and only if the storage matrix depends on time *and* on its previous values. Note that a multi-static strategy is not considered dynamic: independently solving static problems does not make a dynamic strategy. In general, history-based strategies are dynamic because they create an implicit dependency among storage matrices produced at different instants. In [25], static data replication is discussed. The authors show that the grid data replication problem is NP-hard and non-approximable, suggesting that heuristics have to be developed to solve it efficiently. Dynamic replication strategies are described, e.g., in [26].

Among dynamic strategies, infrastructure-adapting strategies are able to adjust changes in the infrastructure charactersitics (e.g. storage element availability, network bandwidths) or usage (e.g. file access patterns) and online strategies are able to adjust changes in the set of considered files. Note that in case of online strategies, the number of rows of the storage matrix may evolve. In case of infrastructure-adapting, the number of columns of the storage matrix may evolve.

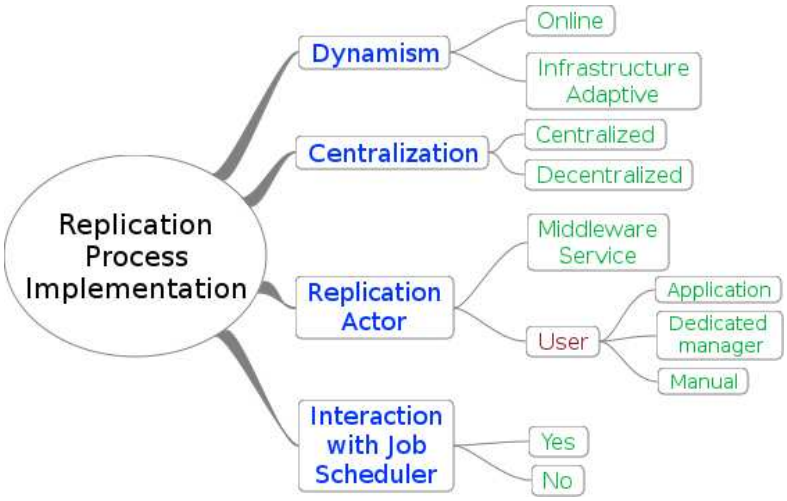

Figure 8: Taxonomy of the replication processes.

.

The replication actor is another characteristic of the replication process. It can belong to the middleware or to the user space. In the former, basic infrastructure services are instrumented for file placement. For instance this is the case in [26] where storage elements decide to hold a file or not based on the profit they can make about it. Placement strategies implemented by the middleware are in principle more efficient because they provide a uniform consistent method across the infrastructure. However they also introduce extra complexity in the middleware design, which may reduce fault tolerance. The user space can be split as applications or dedicated managers. In the former case, the application itself is responsible for taking placement decisions, allowing more specific optimizations. In the latter, middleware overlay services such as workflow managers handle file placement. Most replication strategies, such as described in [21], [27] and [28] are implemented as middleware services.

Finally, there is a distinction between file placement processes that can interact with the job scheduler and those that cannot. Cooperation between data placement and job scheduling can improve the overall transfer time and have a significant impact on the application makespan as shown in [27]. For instance, high-energy physics experiments running on EGI preferably schedule their jobs on the sites where the processed data is located. Another example is shown in [19], where the scheduler (resource broker) communicates with replica optimization services to estimate the job completion time taking data transfer times into account.

4.5. Taxonomy of replication validation methods

Figure 9 shows the taxonomy of validation methods. The validation method gives sound information about the scope of the optimization. In many cases the evaluation is done in simulation, using gold-standard simulators such as OptorSim [29] or ad-hoc

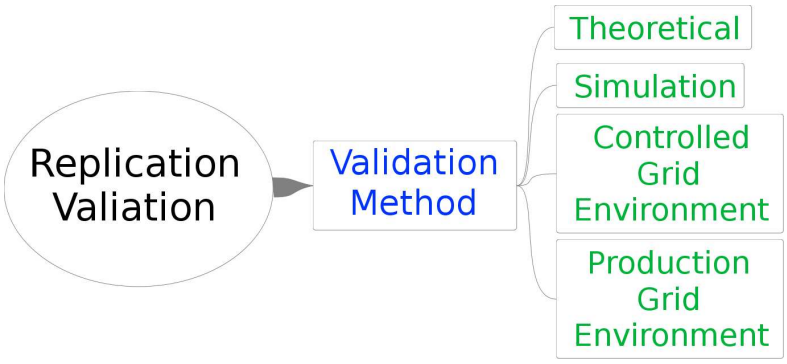

Figure 9: Taxonomy of replication validation.

.

developments. In some cases such as [8] and [30], experiments are implemented on a real infrastructure or controlled environment. Theoretical validation may rely on a mathematical modeling of the problem [8] or on formal proofs of algorithm performance [25].

5. Classification of existing methods

The taxonomies presented above extend the taxonomies of replication architecture and strategy in the survey of data grids presented in [11]: a new validation facet is introduced, clairvoyance predicate is added to the resource model, optimization target is added to the optimization method, replication actor is added to the process implementation, dynamic methods are classified as online and infrastructure-adapting, two new classes are added to the file type predicate and three new classes are added to the resource model.

Tables 1 to 4 present a classification of the reviewed replication methods according to these taxonomies; + / - symbols indicate that the paper belong / does not belong to the class and *?* indicate undetermined classification due to lack of information or irrelevance of the class to the paper (for instance, clairvoyance about computing resources is not relevant for replication processes that do not take computing resources into account). This classification concentrates on the recent works and brings an important update to [11]: 80% of the classified papers were published during the last 6 years.

Online and infrastructure-adaptive approaches are used by the vast majority of works, due to the variability of grid conditions and usage. Most works also consider file replication from the point of view of a middleware service, although manual implementations are seen in production. Few works take into account file modifications.

Tables 2 and 4 present works that were evaluated at production scale. These include the 4 main high-energy physics experiments CMS, ALICE, LHCb and ATLAS. The classification shows a clear distinction between these works and the ones that stay at a theoretical level or are evaluated in simulation or in controlled conditions. The next

Classification				Papers													
Subject/ Facet	Predicate	Object/Class		*Jiang et al.* [31]	*Yang et al.* [32]	*Sato et al.* [8]	*Lei et al.* [20]	*Vrbsky et al.* [30]	*Carman et al.* [28]	*Elghirani et al.* [26]	*Tang et al.* [22]	*Abdurrab et al.* [33]	*Bell et al.* [19]	*Ranganathan et al.* [21]	*Cibej et al.* [25]	*Elghirani et al.* [27]	*Tang et al.* [23]
File Model	File Type	Input / output files are distinguished		-	-	-	-	-	-	-	-	-	-	-	-	-	-
		Temporary / permanent files are distinguished		-	-	-	-	-	-	-	-	-	-	-	-	-	-
		File collections are considered		-	-	-	-	-	-	-	-	-	-	-	-	-	-
	File Access Pattern	Statistical relation		+	+	+	+	+	-	+	+	+	+	+	-	-	+
		Temporal relation		-	-	-	-	-	+	-	-	-	-	+	-	-	-
		Geographical relation		-	-	-	-	-	+	-	-	-	-	+	-	-	-
		Spatial relation		-	-	-	-	-	+	-	-	-	-	+	-	-	-
		Workflow relation		-	-	-	-	+	-	-	-	+	+	-	+	+	-
	File Access Permissions	Modify		-	+	-	-	-	-	-	-	-	-	-	-	-	-
	File Origin	Single source		-	-	-	-	-	-	+	+	+	+	+	?	+	-
		Multiple sources		+	+	+	+	+	+	-	-	-	-	-	?	-	+
Resource Model	Grid Topology	Multiple tier		-	-	-	-	-	-	-	+	+	+	+	?	+	-
	Site Model	SE alone		-	-	+	-	-	-	+	-	+	+	?	?	+	-
		CE alone		-	-	?	-	-	-	+	-	+	+	?	?	-	+
		Combination of SE(s) and CE(s)		+	+	?	+	+	+	+	+	+	+	?	?	+	+
	Clairvoyance	Clairvoyant	Storage resources	+	+	+	+	+	?	+	+	+	+	?	?	+	+
			Compute resources	?	?	?	?	?	?	?	?	+	+	?	?	+	+
			Network resources	+	+	+	-	?	?	+	+	+	+	+	+	+	+

Table 1: Classification of file replication papers (1-1).

Classification				Papers				
Subject/ Facet	Predicate	Object/Class		ALICE [37] [41] [44]	CMS [34] [38] [10] [9]	LHCb [35] [39] [42] [45]	ATLAS [36] [40] [43] [46]	Pegasus and DRS [47]
File Model	File Type	Input / output files are distinguished		+	?	+	+	+
		Temporary / permanent files are distinguished		-	-	-	-	-
		File collections are considered		+	+	+	+	+
	File Access Pattern	Statistical relation		-	-	-	+	-
		Temporal relation		-	-	-	-	-
		Geographical relation		-	-	-	-	-
		Spatial relation		-	-	-	-	-
		Workflow relation		-	-	-	-	+
	File Access Permissions	Modify		-	-	-	+	-
	File Origin	Single source		+	+	+	-	+
		Multiple sources		+	?	+	+	+
Resource Model	Grid Topology	Multiple tier		+	+	+	+	-
	Site Model	SE alone		-	-	-	-	-
		CE alone		-	-	-	-	-
		Combination of SE(s) and CE(s)		+	+	+	+	+
	Clairvoyance	Clairvoyant	Storage resources	+	+	+	+	-
			Compute resources	-	-	-	?	+
			Network resources	-	-	-	-	-

Table 2: Classification of file replication papers (1-2).

Classification			Papers													
Subject/ Facet	Predicate	Object/Class	*Jiang et al.* [31]	*Yang et al.* [32]	*Sato et al.* [8]	*Lei et al.* [20]	*Vrbsky et al.* [30]	*Carman et al.* [28]	*Elghirani et al.* [26]	*Tang et al.* [22]	*Abdurrab et al.* [33]	*Bell et al.* [19]	*Ranganathan et al.* [21]	*Cibej et al.* [25]	*Elghirani et al.* [27]	*Tang et al.* [23]
Replication Process Implementation	Dynamism	Online	-	+	-	-	-	+	-	+	+	+	+	-	+	+
		Infrastructure Adaptive	-	+	+	+	+	+	+	+	+	+	+	-	+	+
	Centralization	Centralized	+	+	+	-	-	-	-	+	-	-	-	?	+	+
	Replication Actor	Middleware service	+	+	+	+	+	+	+	+	+	+	+	?	+	+
		User: Application	-	-	-	-	-	-	-	-	-	-	-	?	-	-
		User: Dedicated manager	-	-	-	-	-	-	-	-	-	-	-	?	-	-
		User: Manual	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Interaction with Job Scheduler	Yes	-	-	-	-	-	-	-	-	-	+	-	-	+	+
Replication Optimization Method	Replication Target	Application	+	+	-	-	-	-	+	-	+	+	+	+	+	+
		Grid Infrastructure	-	+	+	+	+	+	+	+	+	-	+	+	-	+
	Optimization Scope	Local	-	-	+	-	-	+	-	-	+	+	+	-	-	-
		Global	+	+	-	+	+	-	+	+	-	-	+	+	+	+
	Optimization Metric	Availability	-	-	-	+	-	-	-	-	-	-	-	-	-	+
		Transfer Time	+	+	+	-	+	+	-	+	+	+	+	+	+	+
		Cost	-	+	+	-	+	+	+	+	+	-	+	+	-	+
Replication Validation	Validation Method	Theoretical	+	-	+	+	-	+	+	-	-	-	-	+	+	+
		Simulation	+	+	+	+	+	+	+	+	+	+	+	-	+	+
		Controlled grid environment	-	-	+	-	+	-	-	-	-	-	-	-	-	-
		Production grid environment	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 3: Classification of file replication papers (2-1).

Classification				Papers				
Subject/ Facet	Predicate	Object/Class		ALICE [37] [41] [44]	CMS [34] [38] [10] [9]	LHCb [35] [39] [42] [45]	ATLAS [36] [40] [43] [46]	Pegasus and DRS [47]
Replication Process Implementation	Dynamism	Online		+	+	+	+	+
		Infrastructure Adaptive		?	?	?	+	+
	Centralization	Centralized		-	-	-	+	+
	Replication Actor	Middleware service		-	-	-	+	+
		User	Application	-	-	-	-	-
			Dedicated manager	-	-	-	-	-
			Manual	+	+	+	-	+
	Interaction with Job Scheduler	Yes		+	+	+	+	+
Replication Optimization Method	Replication Target	Application		+	+	+	-	+
		Grid Infrastructure		-	-	-	+	-
	Optimization Scope	Local		-	-	-	-	+
		Global		+	+	+	+	-
	Optimization Metric	Availability		+	+	+	+	+
		Transfer Time		+	+	+	+	+
		Cost		-	-	-	+	-
Replication Validation	Validation Method	Theoretical		-	-	-	-	-
		Simulation		-	-	-	-	-
		Controlled grid environment		-	-	-	-	-
		Production grid environment		+	+	+	+	+

Table 4: Classification of file replication papers (2-2).

section discusses differences between theoretical works and the solutions deployed in production.

6. File replication methods in production systems

Most high-performance computing infrastructures (HPC) use parallel file systems which completely hide data management operations. In these file systems, data replication is automatic, but it cannot exploit application-level information such as file type or access pattern. Conversely, high-throughput computing (HTC) infrastructures delegate data management to middleware or application-level services. File replication policies can be defined according to applications' characteristics, but they are still mostly manual. This section details file replication methods used in Teragrid, DEISA/PRACE, OSG and EGI. Although parallel file systems are out of the scope of our taxonomy (they are very different from application-level solutions), we mention them here for the sake of completeness. We then discuss some of the possible reasons explaining the lack of automated replication strategies in HTC systems.

6.1. Data replication in production grids

HPC infrastructures such as Teragrid (now called XSEDE[9]) or in Europe DEISA (now called PRACE[10]) have been using parallel file systems to share data among nodes in a single network domain. For instance, DEISA/PRACE uses an adaptation of IBM's Multicluster General Parallel File System (GPFS [49]), a POSIX-compliant file system striping files in blocks placed on different disks. Block size is adjusted to file and disk size to balance between throughput and disk utilization. To increase data availability, system administrators may define disk failure groups and set a replication factor n; the system then ensures that blocks are replicated n times in different failure groups. The Lustre filesystem is also used in Teragrid/XSEDE. Lustre totally relies on the storage hardware for data replication[11], most likely RAID [50].

Parallel file systems were extended to handle data distributed on wide-area networks (WAN), and many TeraGrid projects have used WAN file-systems in production. The main implementations of WAN file-systems are Lustre-WAN [51] and GPFS-WAN [52]. A few studies proved the performance of WAN filesystems on production grids, see for instance [53] for Lustre. However, replication policies used in these WAN extensions inherits those used in their initial file systems. It means that replication is performed regardless of the application, file type or other user-level characteristics. For data-intensive applications, this has an impact on cost (e.g. data footprint), performance and reliability. An exception is SLASH2[12], a file system for widely distributed systems used on the TeraGrid. In SLASH2, replication is triggered by users, but its consistency and maintenance is performed by the file system.

[9] `https://www.xsede.org`
[10] `http://prace-ri.eu`
[11] `http://wiki.lustre.org`
[12] `http://quipu.psc.teragrid.org/slash2`

Conversely, HTC infrastructures such as Open-Science Grid and the European Grid Infrastructure mostly handle file replication through application-level services. The main replica catalog in the Open-Science Grid (OSG[13]) is the Globus Replica Location Service (RLS [54]), which links logical file names to replicas. While several replicas of a given file can be managed by RLS, it cannot automatically trigger replication based on some policy. In [55], an extension of RLS to distribute the file catalog is described. While this is useful to prevent the catalog to become a single point of failure, this does not concern the optimization of file placement itself. The work in [56] presents a data replication service (DRS) ensuring the reliability of data replication, but no replica placement strategy seems to be available either.

Data management services on the European Grid Infrastructure (EGI[14]) allow to register and replicate files in file catalogs (in particular the LCG File Catalog), but replication decisions are left to the applications. In practice, most applications rely on manual replication, either statically (replication locations are pre-set in configuration files) or dynamically (replication locations are adjusted by operators). For instance, in the ATLAS and CMS high-energy physics experiments, data is pre-placed according to the computing model. To the best of our knowledge, no automated dynamic file replication system is currently used in production on EGI.

A few experimental services, however, are currently being designed to perform automatic, dynamic replication in EGI. The work in [35] shows that dynamic replication improves job throughput and that simple strategies such as LRU and LFU perform well. A service to record file popularity is described in [36], and used in [57] to clean unused replicas. The work in [58] suggests that dynamic data placement could be envisaged from this service, but no real strategy is described yet.

6.2. Discussion

Several reasons, described below, explain this lack of automated, dynamic replication strategy in production infrastructures. First, the file model assumed in theory, simulation and controlled conditions is clearly different than the one considered by production works. While the latter often distinguish input/output data and file collections, the others mostly assume undistinguished files. Conversely, production works hardly rely on elaborated access patterns. ATLAS just started considering file popularity to steer replication [36] but statistical, temporal, geographical and spatial access patterns are not seen in production because such assumptions are often assumed hazardous [46]. Determining file access patterns from the application workflow as done in [47, 59] is more realistic. We conclude that replication policies should rely on *a-priori* information about file accesses, such as file type or workflow relation.

Non-clairvoyance is another important characteristic of production works. While clairvoyance about compute and network resources is almost always assumed in theoretical, simulation and experimental works, Table 2 shows that it is not a valid assumption in production. Clairvoyance about storage resources is realistically implemented

[13] `http://www.opensciencegrid.org`
[14] `http://www.egi.eu`

though. Therefore, there is a need for file replication strategies that do not rely on network throughput, job queuing time or job execution time.

Moreover, file availability is always considered in production works while it is seldom studied by others. Conversely, cost is not studied in the reviewed production works, except in [18]. In fact, most reviewed production works do not experience resource shortage but consider fault-tolerance as a major concern. Designing file replication strategies focusing on file availability is required for production.

7. Conclusion

We presented a formalization of the file placement and replication problem on grids and we proposed a classification of recent approaches based on taxonomies of the file model, the resource model, the replication process, the optimization method and the replication validation. These taxonomies were described using an RDF-like *subject-predicate-object* notation as described in [17]. A total of 45 classes were identified.

A clear gap was identified between production approaches and the ones staying at a theoretical level or evaluated in simulation or in controlled conditions. Significant differences in the file models, in assumptions about the file access pattern, in clairvoyance about the infrastructure and in the studied metrics were highlighted. In practice, replication strategies don't seem to be used in any production system yet, although they were widely studied in theory.

Conclusions from this study will be used to design file management strategies for the Virtual Imaging Platform [60] on the European Grid Infrastructure.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (60777004) and International S&T Cooperation Project of China (2007DFB30320). It is also supported by the French National Grid Initiative, "France-Grilles"[15].

References

[1] F. Gagliardi, B. Jones, F. Grey, M. E. Bégin, and M. Heikkurinen. Building an infrastructure for scientific grid computing: status and goals of the egee project. *Royal Society of London Philosophical Transactions Series A*, 363:1729–1742, aug 2005.

[2] Ruth Pordes, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, Frank Würthwein, Ian Foster, Rob Gardner, Mike Wilde, Alan Blatecky, John McGee, and Rob Quick. The open-science grid. *Journal of Physics: Conference Series*, 78(1):012057, 2007.

[15] `http://www.france-grilles.fr`

[3] O. Smirnova, P. Eerola, T. Ekelöf, M. Ellert, J. Hansen, A. Konstantinov, B. Kónya, J. Nielsen, F. Ould-Saada, and A. Wäänänen. Lecture notes in computer science. In Peter Sloot, David Abramson, Alexander Bogdanov, Jack Dongarra, Albert Zomaya, and Yuriy Gorbachev, editors, *Computational Science - ICCS 2003*, volume 2657, chapter The NorduGrid Architecture and Middleware for Scientific Applications, pages 665–665. Springer Berlin / Heidelberg, 2003.

[4] T. Li, S. Camarasu-Pop, T. Glatard, T. Grenier, and H. Benoit-Cattin. Optimization of mean-shift scale parameters on the egee grid. In *Studies in health technology and informatics, Proceedings of Healthgrid 2010*, volume 159, pages 203–214, 2010.

[5] Matthan W. A. Caan, Frans M. Vos, Antoine H. C. van Kampen, Silvia D. Olabarriaga, and Lucas J. van Vliet. Gridifying a diffusion tensor imaging analysis pipeline. In *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, CCGRID '10, pages 733–738, Washington, DC, USA, 2010. IEEE Computer Society.

[6] Dagmar Krefting, Ralf Luetzkendorf, Kathrin Peter, and Johannes Bernarding. Performance analysis of diffusion tensor imaging in an academic production grid. *Cluster Computing and the Grid, IEEE International Symposium on*, pages 751–756, 2010.

[7] R. Chang, J. Chang, and S. Lin. Job scheduling and data replication on data grids. *Future Generation Computer Systems*, 23(7):846, 2007.

[8] H. Sato, S. Matsuoka, T. Endo, and N. Maruyama. Access-pattern and bandwidth aware file replication algorithm in a grid environment. *9th IEEE/ACM International Conference On Grid Computing*, pages 250–257, 2008.

[9] L. Tuura, B. Bockelman, D. Bonacorsi, R. Egeland, D. Feichtinger, S. Metson, and J. Rehn. Scaling cms data transfer system for lhc start-up. *Journal of Physics: Conference Series*, 119(7):072030, 2008.

[10] J. Rehn, T. Barrass, D. Bonacorsi, J. Hernandez, I. Semeniouk, L. Tuura, and Y. Wu. Phedex high-throughput data transfer management system. In *Computing in High Energy Physics, CHEP'2006*, 2006.

[11] Srikumar Venugopal, Rajkumar Buyya, and Kotagiri Ramamohanarao. A taxonomy of data grids for distributed data sharing, management, and processing. *ACM Comput. Surv.*, 38(1), June 2006.

[12] T. Glatard, C. Lartizien, Bernard Gibaud, R. Ferreira da Silva, G. Forestier, F. Cervenansky, M. Alessandrini, H. Benoit-Cattin, O. Bernard, S. Camarasu-Pop, N. Cerezo, P. Clarysse, Alban Gaignard, Patrick Hugonnard, H. Liebgott, S. Marache, A. Marion, J. Montagnat, J. Tabary, and D. Friboulet. A virtual imaging platform for multi-modality medical image simulation. *IEEE Transactions on Medical Imaging*, in press, 2012.

[13] Ewa Deelman and Ann Chervenak. Data management challenges of data-intensive scientific workflows. In *2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, page 687, 2008.

[14] Philip H. Carns, Walter B. Ligon, III, Robert B. Ross, and Rajeev Thakur. Pvfs: a parallel file system for linux clusters. In *Proceedings of the 4th annual Linux Showcase & Conference - Volume 4*, pages 28–28, Berkeley, CA, USA, 2000. USENIX Association.

[15] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. *Mass Storage Systems and Technologies, IEEE / NASA Goddard Conference on*, 0:1–10, 2010.

[16] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, pages 29–43, New York, NY, USA, 2003. ACM.

[17] Domenico Talia, Ramin Yahyapour, Wolfgang Ziegler, Marek Wieczorek, Andreas Hoheisel, and Radu Prodan. Taxonomies of the multi-criteria grid workflow scheduling problem. In *Grid Middleware and Services*, pages 237–264. Springer US, 2008.

[18] Gurmeet Singh, Karan Vahi, Arun Ramakrishnan, Gaurang Mehta, Ewa Deelman, Henan Zhao, Rizos Sakellariou, Kent Blackburn, Duncan Brown, Stephen Fairhurst, David Meyers, G. Bruce Berriman, John Good, and Daniel S. Katz. Optimizing workflow data footprint. *Sci. Program.*, 15(4):249–268, December 2007.

[19] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini. Evaluation of an economy-based file replication strategy for a data grid. In *3rd IEEE/ACM International Symposium on Cluster Computing and the Grid*, page 661, 2003.

[20] M. Lei, S. V. Vrbsky, and X. Y. Hong. An on-line replication strategy to increase availability in data grids. *Future Generation Computer Systems*, 24(2):85–98, 2008.

[21] Kavitha Ranganathan and Ian Foster. Identifying dynamic replication strategies for a high-performance data grid. In Craig Lee, editor, *Grid Computing —- GRID 2001*, volume 2242, pages 75–86. Springer Berlin / Heidelberg, 2001.

[22] Ming Tang, Bu-Sung Lee, Chai-Kiat Yeo, and Xueyan Tang. Dynamic replication algorithms for the multi-tier data grid. *Future Generation Computer Systems*, 21(5):775, 2005.

[23] Ming Tang, Bu-Sung Lee, Xueyan Tang, and Yeo Chai-Kiat. The impact of data replication on job scheduling performance in the data grid. *Future Generation Computer Systems*, 22(3):254, 2006.

[24] M. M. Deris, J. H. Abawajy, and H. M. Suzuri. An efficient replicated data access approach for large-scale distributed systems. In *IEEE International Symposium on Cluster Computing and the Grid, 2004. CCGrid 2004.*, page 588, 2004.

[25] U. Cibej, B. Slivnik, and B. Robic. The complexity of static data replication in data grids. *Parallel Computing*, 31(8-9):900, 2005.

[26] Ali H. Elghirani, Riky Subrata, and Albert Y. Zomaya. A proactive non-cooperative game-theoretic framework for data replication in data grids. In *2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, page 433, 2008.

[27] Ali Elghirani, Riky Subrata, and Albert Y. Zomaya. Intelligent scheduling and replication in datagrids: a synergistic approach. *IEEE International Symposium on Cluster Computing and the Grid*, pages 179–182, 2007.

[28] M. Carman, F. Zini, L. Serafini, and K. Stockinger. Towards an economy-based optimisation of file access and replication on a data grid. In *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, page 340, 2002.

[29] William H. Bell, David G. Cameron, A. Paul Millar, Luigi Capozza, Kurt Stockinger, and Floriano Zini. Optorsim: A grid simulator for studying dynamic data replication strategies. *International Journal of High Performance Computing Applications*, 17(4):403–416, 2003.

[30] Susan V. Vrbsky, Ming Lei, Karl Smith, and Jeff Byrd. Data replication and power consumption in data grids. In *IEEE International Conference on Cloud Computing Technology and Science*, page 288, 2010.

[31] Jianjin Jiang and Guangwen Yang. An optimal replication strategy for data grid systems. *Frontiers of Computer Science in China*, 1(3):338, 2007.

[32] Chao-Tung Yang, Chun-Pin Fu, and Ching-Hsien Hsu. File replication, maintenance, and consistency management services in data grids. *The Journal of Supercomputing*, 53(3):411, 2010.

[33] Abdul Rahman Abdurrab and Tao Xie. Fire: A file reunion based data replication strategy for data grids. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, page 215, 2010.

[34] R. Egeland, T. Wildish, and S. Metson. Data transfer infrastructure for cms data taking. In *Proceedings of the 12th Advanced Computing and Analysis Techniques in Physics Research*, 2008.

[35] C. Nicholson, D. G. Cameron, A. T. Doyle, A. P. Millar, and K. Stockinger. Dynamic data replication in lcg. *Concurrency and Computation: Practice and Experience*, 20(11):1259, 2008.

[36] A. Molfetas, F. Megino, A. Tykhonov, V. Garonne, S. Campana, M. Lassnig, M. Barisits, G. Dimitrov, and F. Viegas. Popularity framework to process dataset tracers and its application on dynamic replica reduction in the atlas experiment. Technical report, Geneva, Jun 2011.

[37] S. Bagnasco, L. Betev, P. Buncic, F. Carminati, F. Furano, A. Grigoras, C. Grigoras, P. Mendez Lorenzo, A. J. Peters, and P. Saiz. The alice workload management system: Status before the real data taking. *Journal of Physics: Conference Series*, 219(6):062004, 2010.

[38] Alessandra Fanfani, Anzar Afaq, Jose Sanches, Julia Andreeva, Giusepppe Bagliesi, Lothar Bauerdick, Stefano Belforte, Patricia Bittencourt Sampaio, Ken Bloom, Barry Blumenfeld, Daniele Bonacorsi, Chris Brew, Marco Calloni, Daniele Cesini, Mattia Cinquilli, Giuseppe Codispoti, Jorgen D'Hondt, Liang Dong, Danilo Dongiovanni, Giacinto Donvito, David Dykstra, Erik Edelmann, Ricky Egeland, Peter Elmer, Giulio Eulisse, Dave Evans, Federica Fanzago, Fabio Farina, Derek Feichtinger, Ian Fisk, Josep Flix, Claudio Grandi, Yuyi Guo, Kalle Happonen, José Hernàndez, Chih-Hao Huang, Kejing Kang, Edward Karavakis, Matthias Kasemann, Carlos Kavka, Akram Khan, Bockjoo Kim, Jukka Klem, Jesper Koivumäki, Thomas Kress, Peter Kreuzer, Tibor Kurca, Valentin Kuznetsov, Stefano Lacaprara, Kati Lassila-Perini, James Letts, Tomas Lindén, Lee Lueking, Joris Maes, Nicolò Magini, Gerhild Maier, Patricia Mcbride, Simon Metson, Vincenzo Miccio, Sanjay Padhi, Haifeng Pi, Hassen Riahi, Daniel Riley, Paul Rossman, Pablo Saiz, Andrea Sartirana, Andrea Sciabà, Vijay Sekhri, Daniele Spiga, Lassi Tuura, Eric Vaandering, Lukas Vanelderen, Petra Van Mulders, Aresh Vedaee, Ilaria Villella, Eric Wicklund, Tony Wildish, Christoph Wissing, and Frank Würthwein. Distributed analysis in cms. *Journal of Grid Computing*, 8(2):159–179, 2010. 10.1007/s10723-010-9152-1.

[39] A. C. Smith and A. Tsaregorodtsev. Dirac: reliable data management for lhcb. *Journal of Physics: Conference Series*, 119(6):062045, 2008.

[40] M. Branco, D. Cameron, B. Gaidioz, V. Garonne, B. Koblitz, M. Lassnig, R. Rocha, P. Salgado, and T. Wenaus. Managing atlas data on a petabyte-scale with dq2. *Journal of Physics: Conference Series*, 119(6):062017, 2008.

[41] R. Divià, U. Fuchs, I. Makhlyueva, P. Vande Vyvre, V. Altini, F. Carena, W. Carena, S. Chapeland, V. Chibante Barroso, F. Costa, F. Roukoutakis, K. Schossmaier, C. Soòs, B. von Haller, and the ALICE collaboration. The alice online data storage system. *Journal of Physics: Conference Series*, 219(5):052002, 2010.

[42] A. C. Smith and A. Tsaregorodtsev. Dirac: data production management. *Journal of Physics: Conference Series*, 119(6):062046, 2008.

[43] Johannes Elmsheuser, Frederic Brochu, Greig Cowan, Ulrik Egede, Benjamin Gaidioz, Hurng-Chun Lee, Andrew Maier, Jakub Móscicki, Katarina Pajchel, Will Reece, Bjorn Samset, Mark Slater, Alexander Soroko, Daniel Vanderster,

and Michael Williams. Distributed analysis in atlas using ganga. *Journal of Physics: Conference Series*, 219(7):072002, 2010.

[44] S. Bagnasco, L. Betev, P. Buncic, F. Carminati, C. Cirstoiu, C. Grigoras, A. Hayrapetyan, A. Harutyunyan, A. J. Peters, and P. Saiz. Alien: Alice environment on the grid. *Journal of Physics: Conference Series*, 119(6):062012, 2008.

[45] F. Bonifazi, A. Carbone, E. D. Perez, A. D'Apice, L. dell'Agnello, D. Duellmann, M. Girone, G. L. Re, B. Martelli, G. Peco, P. P. Ricci, V. Sapunenko, V. Vagnoni, and D. Vitlacil. Lhcb experience with lfc replication. *Journal of Physics: Conference Series*, 119(4):042005, 2008.

[46] Mario Lassnig, Vincent Garonne, Miguel Branco, and Angelos Molfetas. Dynamic and adaptive data-management in atlas. *Journal of Physics: Conference Series*, 219(6):062054, 2010.

[47] Ann Chervenak, Ewa Deelman, Miron Livny, Mei-Hui Su, Rob Schuler, Shishir Bharathi, Gaurang Mehta, and Karan Vahi. Data placement for scientific applications in distributed environments. In *2007 8th IEEE/ACM International Conference on Grid Computing*, page 267, 2007.

[48] E. Deelman, G. Singh, M.H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G.B. Berriman, J. Good, et al. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, 13(3):219–237, 2005.

[49] Frank Schmuck and Roger Haskin. Gpfs: a shared-disk file system for large computing clusters. In *Proceedings of the 1st USENIX conference on File and storage technologies*, FAST'02, pages 16–16, Berkeley, CA, USA, 2002. USENIX Association.

[50] David A. Patterson, Garth Gibson, and Randy H. Katz. A case for redundant arrays of inexpensive disks (raid). *SIGMOD Rec.*, 17(3):109–116, June 1988.

[51] Joshua Walgenbach, Stephen C. Simms, Kit Westneat, and Justin P. Miller. Enabling lustre wan for production use on the teragrid: a lightweight uid mapping scheme. In *Proceedings of the 2010 TeraGrid Conference*, TG '10, pages 19:1–19:6, New York, NY, USA, 2010. ACM.

[52] Phil Andrews, Patricia Kovatch, and Christopher Jordan. Massive high-performance global file systems for grid computing. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, SC '05, pages 53–, Washington, DC, USA, 2005. IEEE Computer Society.

[53] S. C. Simms, G. G. Pike, and D. Balog. Wide area filesystem performance using lustre on the teragrid. In *TeraGrid Conference*, 2007.

[54] A. L. Chervenak, R. Schuler, M. Ripeanu, M. Ali Amer, S. Bharathi, I. Foster, A. Iamnitchi, and C. Kesselman. The globus replica location service: Design and experience. *IEEE Transactions on Parallel and Distributed Systems*, 20(9):1260, 2009.

[55] Min Cai, Ann Chervenak, and Martin Frank. A peer-to-peer replica location service based on a distributed hash table. In *Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, SC '04, page 56, Washington, DC, USA, 2004. IEEE Computer Society.

[56] Ann Chervenak, Robert Schuler, Carl Kesselman, Scott Koranda, and Brian Moe. Wide area data replication for scientific collaborations. *International Journal of High Performance Computing and Networking*, 5(3):124, oct 2008.

[57] M. Girone, J. Andreeva, F. H. Barreiro Megino, S. Campana, M. Cinquilli, A. Di Girolamo, M. Dimou, D. Giordano, E. Karavakis, M. J. Kenyon, L. Kokozkiewicz, E. Lanciotti, M. Litmaath, N. Magini, G. Negri, S. Roiser, P. Saiz, M. D. Saiz Santos, J. Schovancova, A. Sciabà, D. Spiga, R. Trentadue, D. Tuckett, A. Valassi, D. C. Van der Ster, and J. D. Shiers. The "common solutions" strategy of the experiment support group at cern for the lhc experiments. *Journal of Physics: Conference Series*, 396(3):032048, 2012.

[58] F. H. Barreiro Megino, M. Cinquilli, D. Giordano, E. Karavakis, M. Girone, N. Magini, V. Mancinelli, and D. Spiga. Implementing data placement strategies for the cms experiment based on a popularity model. *Journal of Physics: Conference Series*, 396(3):032047, 2012.

[59] Shishir Bharathi and Ann Chervenak. Data staging strategies and their impact on the execution of scientific workflows. In *Proceedings of the second international workshop on Data-aware distributed computing*, DADC '09, New York, NY, USA, 2009. ACM.

[60] Rafael Ferreira da Silva, Sorina Camarasu-Pop, Baptiste Grenier, Vanessa Hamar, David Manset, Johan Montagnat, Jérôme Revillard, Javier Rojas Balderrama, Andrei Tsaregorodtsev, and Tristan Glatard. Multi-platform workflow execution for medical simulation. In *9th HealthGrid conference*, Bristol, UK, 2011.