Job Scheduling and Data Replication in Hierarchical Data Grid

Somayeh Abdi[1] and Sayyed Mohsen Hashemi[2]

[*1]*Department of Engineering, Eslamabad-E-Gharb Branch, Islamic Azad University, Eslamabde Gharb, Kermanshah, Iran*

[2]*Department of Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*

somayeh.abdi@gmail.com, hashemi@srbiau.ac.ir

Abstract

Data Grid environment is a geographically distributed that deal with date-intensive application in scientific and enterprise computing. In data-intensive applications data transfer is a primary cause of job execution delay. Data access time depends on bandwidth, especially when hierarchy of bandwidth appears in network. Effective job scheduling can reduce data transfer time by considering hierarchy of bandwidth and also dispatching a job to where the needed data are present. Additionally, replication of data from primary repositories to other locations can be an important optimization step to reduce the frequency of remote data access. Objective of dynamic replica strategies is reducing file access time which leads to reducing job runtime. In this paper we develop a job scheduling policy, called SS (Scheduling strategy), and a dynamic data replication strategy, called DRS (Distributed Replication Strategy), to improve the data access efficiencies in a cluster grid. We study our approach and evaluate it through simulation. The results show that combination of SS and DRS has improved 17% over other combinations.

Keywords*: Data Grid, Job scheduling, Data Replication, Replica catalogue, Replica Manager*

1. Introduction

In the increasing demand of scientific and large-scale business application, a large amount of data are generated and spread for using by users around the world. Such data cannot be stored centralized but distributed among centers around the world for processing. Motivation by an integrate architecture for storage, data management and data intensive application execution, Data Grid is proposed as a solution. A Grid is a distributed collection of computers and storage resources maintained to enable resource sharing and coordinated problem solving in dynamic, multi-institutional virtual organizations [1]. The Grid research field can be divided into two large sub-domains: Computational Grid and Data Grid. A Computational Grid is a hardware and software infrastructure that provides dependable, consistent and inexpensive access to high-end computational capabilities [1]. Data Grid is an integrating architecture that allows connecting a collection of hundreds of geographically distributed computers and storage resources located in different part of the world to facilitate data and resources sharing [2]. Scientific applications such as data analysis in High Energy Physics (HEP), climate modeling or earth observation are very data intensive and a large community of researchers around the world wants to have fast access to the data. In this application size of data needs to be accessed may be up to peta-bytes that leads to many challenges in Data Grid. Data-intensive applications are one major kind of jobs in Data Grid, and their

scheduling strategy and data management are regarded as the most important research fields [2]. Data management includes replication and movement of data at selected sites. As jobs are data intensive, data management issues often become integral to the problems of scheduling and effective resource management in the Data Grids. The Grid scheduler is one of the most critical components of the resource management systems, since it has the responsibility of assigning resources to jobs by considering the application requirements and resources usage performance.

The Data Grid scheduling approach has three phases: resource discovery, system selection and job execution. Resource discovery identify resources that can be used with their capability when the Data Grid scheduler has to find a suitable set of resources for job execution. One of the aspects that must be considered is how the scheduling efficiently work with the amount of data need for each job and the impact of replication mechanism to the scheduling performance [1]. In data-intensive applications, locations of required data by the job impact the system selection and performance greatly [3]. In large-scale data-intensive applications, data transfer time is the primary cause of job execution delay. Therefore, integration of data replication and scheduling strategies is important [4]. The replication mechanism determines which file should be replicated, when to create new replicas and where the new replicas should be placed. Replication methods can be classified as static and dynamic. For the static replication methods, a replica can persist until it is deleted by users or its duration is expired. On the contrary, dynamic replication takes into consideration the changes of the Grid environment, and it automatically creates new replicas for popular data files or moves the replicas to other sites when is needed to improve the performance [5]. In this paper we develop new scheduling and replication algorithms that reduce data transfer time and job execution time. Our new scheduling policy considers the locations of required data, hierarchy of bandwidth and the capacity of computing nodes. The proposed replication strategy considers bandwidth as an important factor for replica selection and replica placement. It also increases the chances of accessing data at a nearby node in cluster grid.

2. Related Work

There are some recent works that address the problem of scheduling and/ or replication in Data Grid as well as the combination between them. In [3], it considers two centralized and decentralized replication algorithms. In centralized method, replica master uses a table that ranks each file access in descending order. If a file access is less than the average, it will be removed from the table. Then it pop files from top and replicate using a response-time oriented replica placement algorithm. In the decentralized method, every site records file access in its table and exchange this table with neighbors. Since every domain knows average number of access for each file and then deletes those files whose access is less than the average, and replicates other files in its local storage.

In [4], an algorithm for a 2-level hierarchical structure based on internet hierarchy (BHR) has been introduced which only considers dynamic replication and does not consider scheduling. Nodes in the first level are connected to each other with high speed networks and in the second level via internet. The algorithm replicates the file to the site if there is enough space. Next it, accesses the file remotely if the file is available in the sites that are in the same region. Otherwise it tries to make available space by deleting files using LRU (Least Recently Used) method, and replicates the file. It assumes that master site always has a safe copy of file before deleting. In [5], an algorithm for a two-level hierarchical structure based on internet hierarchy (BHR) has been introduced which only considers dynamic replication and

does not consider scheduling. Nodes in the first level are connected to each other with high speed networks and in the second level via internet. The algorithm replicates the file to the site if there is enough space. Next it, accesses the file remotely if the file is available in the sites that are in the same region. Otherwise it tries to make available space by deleting files using LRU (Least Recently Used) method, and replicates the file. It assumes that master site always has a safe copy of file before deleting.

3. The Proposed Method

In this section, we present hierarchical network structure and propose two strategies for job scheduling and data replication base on the hierarchical network structure.

3.1. Network Structure

The Data Grid architecture supporting data replication and job scheduling is shown in Figure 1. In this structure we define a cluster as an organization unit which is a group of sites that are geographically close to each other, each cluster comprises the computers which are connected by a high bandwidth. We define two kinds of communications between sites in this structure, Inter-communication and intra-communication. Intra-communication is the communication between sites within the same cluster and inter-communication is the communication between sites across clusters. Network bandwidth between sites within a cluster will be larger than across clusters. In communication networks, the performance of data-intensive application is underlying available network bandwidth and data access latency, especially in networks that hierarchy of bandwidth appears. Therefore, to reduce access latency and to avoid WAN bandwidth bottleneck in a cluster grid, it is important to reduce the number of inter-communications. Data Grid Information Service (DGIS) providing resource registration services and keeping track of a list of resources available in the Data Grid. Grid Scheduler can query DGIS for resource contact, configuration, and status information. Resource discovery identify resources that can be used with their capability through DGIS. For each file, Replica Catalogue (RC) keeping track of a list of sites that store the replica at each cluster .In hierarchical networks it is important to schedule jobs to cluster (and site) that the most of required data are present. The Grid Scheduler query RC at each cluster for the location of required data by the job's execution, it leads to reduce the number of inter-communications and intra-communications. In proposed structure jobs that entering Data Grid environment submit to Grid scheduler and for each job the Grid scheduler, based on the job scheduling strategy and information that gets from RC and DGIS schedules the job to best cluster and site respectively and then submits the job to local scheduler(LS) in the selected site. In this structure we apply distributed dynamic data replication infrastructure. In this structure we apply distributed replication, when a job is assigned to LS Replica Manager (RM) at site manages the data movement between sites.

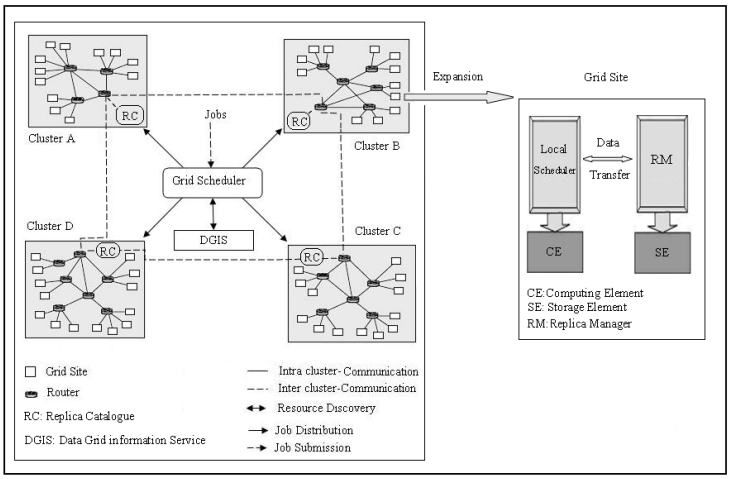

Figure 1. Data Grid Architecture

3.2. Scheduling Strategy

In data-intensive applications, data transfer time is the primary cause of job execution delay and the locations of required data by the job impact the Grid scheduling decision and performance. Our Scheduling Strategy (SS) considers the locations of required data at cluster level and site level in job scheduling decision. The SS determines the best cluster and site respectively and then submit job to Local Scheduler in selected site. For each job execution a best cluster (and site) is a cluster that holds most of the requested data (from size point of view) by the job. This will significantly reduce total transfer time, and reduce the number of inter-communications. Then the job submitted to best site in best cluster, this will reduce the number of intra-communications. Also by placing local RC at each cluster the job scheduling decision time is reduced. In this structure instead of computing the size of required data at each site in the cluster by applying RC that hold list of all data in cluster we could reduce the overload computing time to finding the best cluster in the Data Grid. When the job is accepted to Data Grid environment the Grid Scheduler sends required data by the job to all RCs in clusters. Each RC by comparing the required data with the list of available data in the cluster, computes the size of available data and sends the result to Grid Scheduler. The Grid Scheduler selects the cluster that has the most required data (from size point of view) for job execution. Then Scheduling Strategy select best site in the best cluster for job execution.

Also we assume that $S_{S,c}$ is total size of the requested files available in site S and cluster C, C_C is total size of the requested files available in cluster C and the required data to execute job j are represented as: Rj = {LFN1, LFN2,…., LFNn} and $|LFN_i|$ presents size of file LFNi.

$$S_{S,C} = \sum_{for\ all\ available\ LFNi\ in\ site\ S} |LFN_i| \quad (1)$$

Additionally we define RelativeLoadi that presents the relative load of site i (based on number of jobs at site i and computing capacity of site i).

$$RelativeLoad_i = SizeofJobs_i / C_i \quad (2)$$

Where N is number of sites in cluster C, C_i is computing capacity (in MIPS[1]) of site i and SizeofJobsi is the lengths of queued jobs (in MIPS) on site i. For scheduling each job the SS can be summarized as follow:

1. Grid Scheduler queries the total size of required data at clusters by sending list of required files to RC components at clusters and these components improve scheduling by Computing and sending CC to Grid Scheduler.

2. Grid Scheduler Selects the best cluster $C_{max} = \underset{i=1}{\overset{q}{MAX}} C_i$; where q is the number of cluster (i.e. cluster with most available requested data files)

3. for each site in $C_{\max}$ compute $S_{S,max}$ from(1) by querying the RC at $C_{\max}$

4. Select the best site $S_{\max,\max} = \underset{k=1}{\overset{r}{Max}} S_{k,\max}$; where r is the number of sites in cluster $C_{\max}$ (i.e. Site with most available requested data files from size point of view in cluster $C_{\max}$).

5. If there are several sites with most available data in the $C_{\max}$ Grid Scheduler selects the site with minimum relative load from (2).

When a job is assigned to LS, the RM will be responsible for transferring all the required data by the job that is not available in local site.

3.3. Distributed Replication Strategy

After a job is scheduled to Sj, the requested data will be transferred to Sj to become replicas. DRS (distributed replication Strategy) determine which replica will be transferred to Sj and how to handle this new replica, as shown in Figure 2. DRS consider the inter-communication bandwidth as the main factor for replica selection / deletion. For each required file for job executing, RM controls if required files exit in local site or not; if file doesn't exist in the local site DRS first searches the file in local cluster. If the file duplicated in the same cluster, then it will creates a list of candidate replicas and selects a replica with the maximum bandwidth available for transferring it. If there is enough space for new replica, then it will be stored in local site, otherwise it is only stored in the temporary buffer and will be deleted after the job completes.

If the file doesn't exist in the same cluster, then DRS creates a list of replicas in other clusters and selects the replica with the maximum bandwidth available for transferring it. If there is enough space for new replica, then it will be stored in local site, otherwise occupied space will be released to make enough free space for this new replica. First, it removes the replicas that already exist in other sites in the same cluster based on LRU (least recently used) replacement algorithm. After all these replicas are deleted, if the space is still insufficient, the DRS uses LRU replacement algorithm to delete replica in local storage which is duplicated in other clusters, till it has enough free space for the new replica. To conclude, DRS consider inter-cluster replica transference as very costly. Therefore, the successfully received replicas must be stored locally such that all other sites in the same cluster will access replica with intra-communication way. It leads to reduce the number of inter-communication and reduce data access time.

[1]. Million Instruction Per Second

4. Simulations

Gridsim is used as the simulation tool to evaluate the performance of the proposed replication and scheduling algorithms. The Java-based GridSim discrete event simulation toolkit provides Java classes that represent entities essential for application, resource modeling, scheduling of jobs to resources, and their execution along with management [9]. Its Java-based design makes it portable and available on all computational platforms. The components of Gridsim are as follow and also depicted in Figure 2.

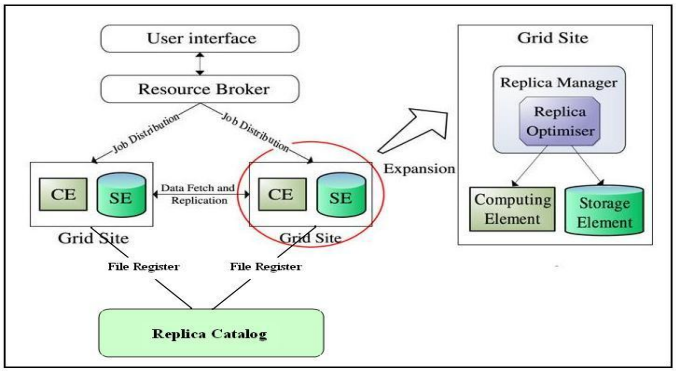

Figure 2. Simulator Architecture

Resource Broker: It receives jobs from user and sends them to the best node according to proposed algorithm. Storage Element (SE),Computing Element (CE), Replica Manager that controls data transferring in each node and provide a mechanism for accessing the Replica Catalogue. Replica Catalogue; This component acts as a centralized RC. It is responsible for indexing available files on the resources and handles queries from users and resources about location of replicas. When each site store a new replica, send a file register request to RC and then RC add this site to the list of sites that holds the replica. Although, we apply this component to each clusters. So these components act as distributed RCs. When RM stores a new replica in the site, send a file register request to local RC at cluster and then RC add this site to the list of sites that holds the replica. Also when RM deletes a replica in the site, send a file delete request to local RC at cluster and then RC deletes this site from list of sites that holds the replica. Replica Optimizer (RO); It has replica algorithm and control file replication according to proposed algorithm. Based on the scheduling algorithm the broker sends jobs to a site. Each job needs a list of files to run. Reducing file access time is the final objective of optimization algorithms.

4.1. Simulation Environment

There are four clusters in our configuration and each cluster has an average of 13 sites, which all have CE with associated SE. Table1 specifies the simulation parameters used in our study. There are 5 job types; each job type requires 12 files to execute. While running, jobs were randomly picked from 5 job types, and then submitted to the Resource Broker. Files are accessed sequentially within a job without any access pattern. To simplify the requirements, data replication approaches in Data Grid environments commonly assume that the data is read-only.

Table 1. Simulation Parameters

Topology Parameters	value
Number of cluster	4
Number of sites in each cluster	13
Storage space at each site	10 GB
Connectivity bandwidth(WAN)	10 Mbps
Connectivity bandwidth(LAN)	1000Mbps
Job parameters	**value**
Number of jobs	500
Number of job types	5
Number of file accessed per job	12
Size of single file	500 MB
Total size of files	50GB

4.2. Simulation Results and Discussion

DRS will be compared with LRU (Least Recently Used) and BHR (Bandwidth Hierarchy based Replication). The LRU algorithm always replicates and then deletes those files that have been used least recently. Figure 3 showed the Average job time based on changing number of jobs for 3 algorithms. The difference between DRS and BHR is that required replica within the same cluster is always the top priority used in DRS, while BHR searches all sites to find the best replica and has no distinction between intra-cluster and inter-cluster. It could be anticipated that DRS will avoid inter-cluster-communications and be stable in hierarchical network architecture with variable bandwidth. Our method takes benefit from network level locality of BHR. Thus, total job execution time is about 12% faster using DRS optimizer than BHR.

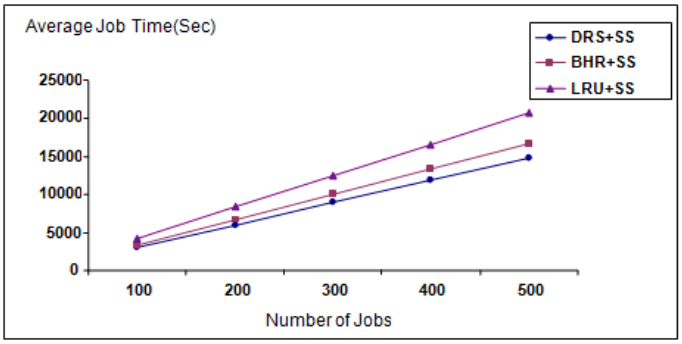

Figure 3. Average Job Time based on Varying Number of Jobs

In the SS the job execution time is the Max {file transmission time, queue time} plus job processing time. SS will be compared with DPRL and Relative Load scheduling strategies. DPRL (Data Present Relative Load) searches all sites to find available CE by using a combination of Data present for the files and the Relative load of sites. Figure 4 showed the Average job time based on changing number of jobs for 3 algorithms. Since SS schedules jobs to certain specific cluster and specific sites according to requested data files. Therefore, jobs would be executed on a cluster with the most needed files. Since the file transmission time is the most important factor to influence the job execution time for data-intensive jobs in Data Grids, SS with DRS can reduce the file transmission time effectively by virtue of valid scheduling and proper data replication.

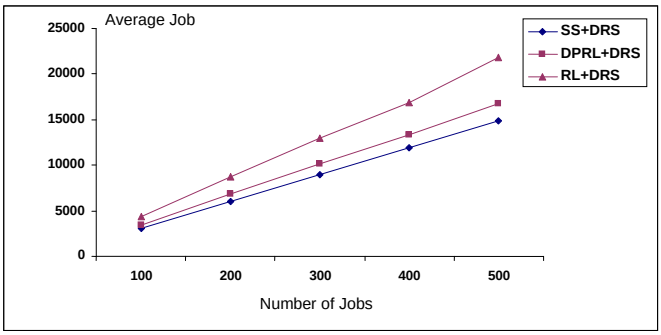

Figure 4. Average Job Time based on Varying Number of Jobs

To analyze the distribution of jobs, we run a simulation where there is a Grid system with four clusters. Each cluster has three grid sites and 500 jobs. Figure 5 showed the distribution of where jobs are executed. Since SS schedule jobs to certain specific sites and specific cluster according to requested data files. Therefore, jobs would be executed on a cluster with the most needed files. It can be observed that the same type of jobs is almost executed at the same cluster as shown in Figure 5(a). On the contrary, DPRL does not take into consideration cluster information and it only schedules jobs to certain specific site, therefore different job types would be executed on a cluster and the number of inter-communication would be increased. It will lead to more overhead in transferring file replicas. The job distribution of DPRL is shown in Figure 5(b).

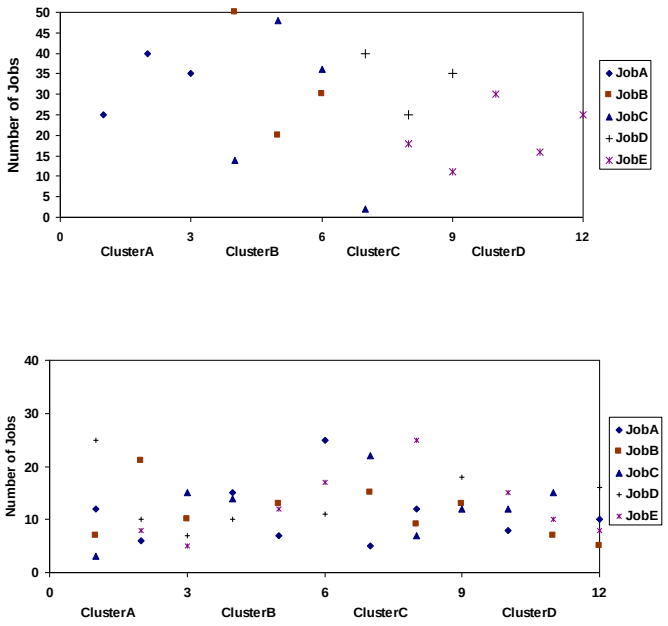

Figure 5. 500 Job Distribution (a) SS with DRS (b) DPRL with LRU

The average number of inter-communications for a job execution is illustrated in Figure 6. By selecting the best cluster and best site, SS with DRS can decrease the number of inter-communications effectively. Overall the simulation results with Gridsim show better performance (over 17%) comparing to current algorithms.

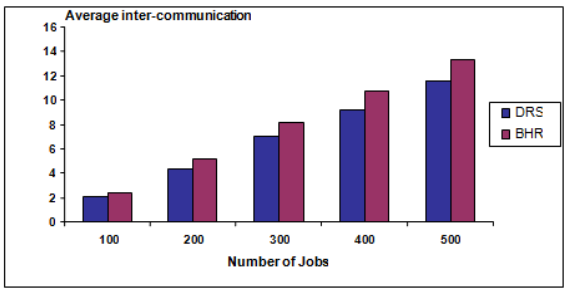

Figure 6. Average number of inter-communications

Figures 7 showed the average job time for 500 jobs. We compare DRS, BHR and LRU algorithms for varying inter-communication bandwidth.

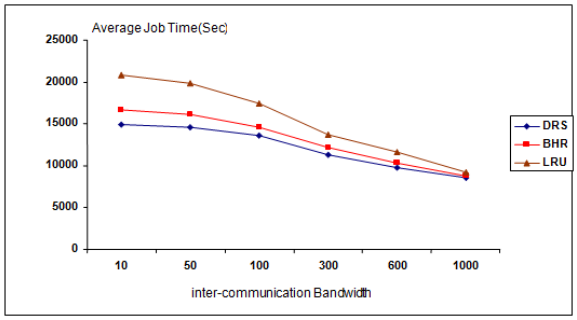

Figure 7. Average Jobs Time with Varying Inter-communication Bandwidth for 500 Jobs

As inter-communication bandwidth increase 3 mentioned algorithms will converge. We can conclude that DRS strategy can be effectively utilized when hierarchy of bandwidth appears.

5. Conclusion and Future Work

In this paper a hierarchical structure for dynamic replicating file and cluster scheduling in Data Grids was proposed. To achieve good network bandwidth utilization and reduce data access time, we propose a job scheduling policy (SS) that considers not only computational capability, job type and data location but also it considers cluster information in job placement decision. We study and evaluate the performance of various replica strategies and different scheduling algorithm combinations. The simulation results show, first of all, that SS and DRS both get better performances than other scheduling policy and replica strategies. Second, we can achieve particularly good performance with SS where jobs are always scheduled to cluster with most of the needed data, and a separate DRS process at each site for replication management. Experimental data showed SS scheduling with DRS replica strategy outperforms others combinations in total job execution time.

Acknowledgements

This research is pecuniary supported by "Eslamabad-E-Gharb Branch, Islamic Azad University".

References

[1] J. Jiang and H. Ji, "scheduling algorithm with potential behaviors", Journal of Computers, vol. 3, no. 12, **(2008)** December.
[2] M. Tang, B.-S. Lee, X. Tang and C. -K. Yeo, "The Impact of Data Replication on Job Scheduling Performance in the Data Grid", Future Generation Computer Systems, vol. 22, Issue 3, **(2006)** February, pp. 254-268.
[3] A. Elghirani, R. Subrata, A. Y. Zomaya and A. Al Mazari, "Performance Enhancement through Hybrid Replication and Genetic Algorithm Co-Scheduling in Data Grids", Advanced Networks Research Group, School of Information Technologies, University of Sydney, NSW, **(2006)**, Australia.
[4] S. -M. Park, J. -H. Kim and Y. -B. Ko, "Dynamic Grid Replication Strategy based on Internet Hierarchy", Book Series Lecture Notes in Computer Science, Grid and Cooperative computing book, Publisher Springer, vol. 3033, **(2005)** August, pp. 838-846.
[5] R. -S. Chang, J. -S. Chang and S. -Y. Lin, "Job scheduling and data replication on Data Grids", Future Generation Computer Systems, vol. 23, Issue 7, **(2007)** August, pp. 846-860.
[6] N. N. Dang and S. B. Lim, "Combination of Replication and Scheduling in Data Grids", IJCSNS International Journal of Computer Science and Network Security, vol. 7, no. 3, **(2007)** March.
[7] T. Phan, K. Ranganathan and R. Sion, "Evolving toward the perfect schedule: Co-scheduling job assignments and data replication in wide-area systems using a genetic algorithm", Job scheduling strategies for parallel processing (11th international workshop), JSSPP 2005, Cambridge MA, **(2005)**.
[8] K. Krauter, R. Buyya and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing", SOFTWARE—PRACTICE AND EXPERIENCE Softw. Pract. Exper., vol. 32, **(2002)**, pp. 135–164 (DOI: 10.1002/spe.432).
[9] K. Krauter and M. Murshed, "GridSim a toolkit for the modelling and simulation of distributed resource management and scheduling for Grid computing", CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE Concurrency Computat.: Pract. Exper., vol. 14, **(2002)**, pp. 1175–1220 (DOI: 10.1002/cpe.710).
[10] The Data Grid Project, http://www.eu-datagrid.org.
[11] Parallel workload Project, http://www.parrallelworkload.org.
[12] The European Data Grid project.
[13] W. H. Bell, D. G. Cameron, L. Capozza, P. Millar, K. Stockinger and F. Zini, "Simulation of dynamic grid replication strategies in OptorSim", in: Proceedings of the Third ACM/IEEE International Workshop on Grid Computing, Grid2002, Baltimore, USA, in: Lecture Notes in Computer Science, vol. 2536, **(2002)**, pp. 46–57.
[14] E. Deelman, H. Lamehamedi, B. Szymanski and S. Zujun, "Data replication strategies in grid environments", in: Proceedings of 5th International Conference on Algorithms and Architecture for Parallel Processing, ICA3PP'2002, IEEE Computer Science Press, Bejing, China, **(2002)**, pp. 378–383.
[15] H. H. Mohamed and D. H. J. Epema, "An evaluation of the close-to-files processor and data co-allocation policy in multiclusters", in: 2004 IEEE International Conference on Cluster Computing, IEEE Society Press, San Diego, California, USA, **(2004)**, pp. 287–298.
[16] M. Carman, F. Zini, L. Serafini and K. Stockinger, "Towards an economybased optimisation of file access and replication on a Data Grid", in: Proceedings of 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGrid 2002, IEEE-CS Press, Berlin, Germany, **(2002)**, pp. 340–345.
[17] P. Kunszt, E. Laure, H. Stockinger and K. Stockinger, "Advanced replica management with reptor", in: Proceedings of 5th International Conference on Parallel Processing and Applied Mathemetics, PPAM 2003, Czestochowa, Poland, **(2003)** September, pp. 848–855.
[18] D. G. Cameron, A. P. Millar and C. Nicholson, "OptorSim: A simulation tool for scheduling and replica optimisation in Data Grids", in: Proceedings of Computing in High Energy Physics, CHEP 2004, Interlaken, Switzerland, **(2004)** September.
[19] K. Ranganathan and I. Foster, "Identifying Dynamic Replication Strategies for a High Performance Data Grid", In Proceedings of the International Grid Computing Workshop, Denver, Colorado, USA, **(2001)**.
[20] I. Foster and K. Ranganathan, "Design and evaluation of dynamic replication strategies for high performance Data Grids", in: Proceedings of International Conference on Computing in High Energy and Nuclear Physics, Beijing, China, **(2001)** September.