IMPROVEMENT OF JOB SCHEDULING AND TOW LEVEL DATA REPLICATION STRATEGIES IN DATA GRID

Fatemeh Jolfaei[1] and Abolfazl T. Haghighat[2]

[1]Department of Electrical, Computer and IT Engineering, Qazvin branch Islamic Azad University Qazvin, Iran
Fatemeh_jolfaei@yahoo.com
[2]Department of Electrical, Computer and IT Engineering, Qazvin branch Islamic Azad University Qazvin, Iran
haghighat@qiau.ac.ir

ABSTRACT

With the development of grid computing, Data Grid, as an important branch of Grid Computing focuses on supporting an efficient management mechanism for controlled sharing and large amounts of distributed data. Data intensive jobs are one major kind of jobs in Data Grid, and their scheduling strategies are regarded as one of the most important research fields. Dynamic replication in data grid aims to reduce data access time and to utilize network and storage resources efficiently. This paper proposes a novel dynamic replication strategy, called ERS(Enhancement of Replication Strategy), which reduces data access time by avoiding network congestions in a data grid network and we develop a job scheduling policy, called ESS (Enhancement of Scheduling strategy).

KEYWORDS

Data Grids, Scheduling, Data Replication, Data Access Time, Execution Time, GridSim.

1.INTRODUCTION

Today's scientific as well as industrial world requires processing of vast amount of distributed data [14]. Grid was first proposed by Foster and Kesselman [1], which is a mechanism for resource sharing and problem solving in heterogeneous environment. An important kind of grid is data grid, which is an integrating architecture that allow connect a collection of hundreds of geographically distributed computers and storage resources located in different part of the world to facilitate sharing of data and resources[8], [14]. In data grids [24], [25], distributed scientific and engineering applications often require access to a large amount of data (terabytes or petabytes). Managing this large amount of data in a centralized way is ineffective due to extensive access latency and load on the central server. Hence, such huge dataset must be separated and stored in several physical locations. In a communication environment, the

performance of accessing a distributed and huge amount of data depends on the availability of network bandwidth. Namely, slow data access can throttle the performance of data-intensive applications running on grid computers. Many examples can be listed such as meteorology, High Energy Physics, computational genomics which processed and resulted large amount of data.
The major obstacles that degrade performance of data grid are latency of wide area networks and the bandwidth of internet[7]. Therefore, latency of network and bandwidth between storage sites and processing sites plays an important role in data grid. Size of data that needs to be accessed on the Data Grid may be upto petabytes in the near future. The nature of dealing with large amount of data that geographically spread causes may challenges in Data Grid. One of them is how the scheduling efficiently work with the amount of data need for each job and the impact of replication to the scheduling performance. Replication is an effective mechanism to reduce file transfer time and bandwidth consumption in data grids[5], [6].

Replication and scheduling problem has been studied separately for long time, however those in Data Grid have just recently received attention from researchers[9]. We proposed a new job scheduling algorithm and data replication policy that reduce job execution time by reducing job data access time. Our new scheduling policy considers the locations of required data and the job queue length of a computing node. We develop a replication strategy, called ERS (Enhancement of Replication Strategy). It takes into account bandwidth as an important factor for replica selection and replica placement. It also increases the chances of accessing data at a nearby node and uses one mechanism for prevention of full storage.

The rest of the paper is organized as follows. Related work on replication and scheduling is given in section 2. Section 3 will include the proposed strategy. Experimental results are shown in section 4. Finally conclusion and future work is given.

2. RELATED WORKS

Dynamic replication is a long-term optimization technique which aims at reducing average job execution time in data grid. Since very large quantity of data files are developed in data grid, there will be certain limitation of amount of files which can be stored at each site. If SE (Storage Element) at a grid site is already filled up whit replicas, some of them should be deleted in order to store newly requested data [13]. There are some recent works that address the problem of scheduling and/or replication in Data Grid as well as the combination between them. An initial work on dynamic data replication in grid was done by [26], [3] proposing six replication strategies: No Replication, Best Client, Cascading, Plain Cascading, Cascading plus Cashing and Fast Spread. The analysis reveals that among all these, Fast Spread shows a relatively consistent performance through various access patterns.

The realization of importance of data locality in job scheduling problem was first proposed by Ranganathan, K. and Foster, I. [26]. The authors had proposed a Data Grid architecture based on 3 main components: External Scheduler (ES), Local Scheduler (LS) and Dataset Scheduler (DS). ES receives job submission from user, and then it decides which remote site to which to send the job depends on ES’s scheduling strategy. LS of each site decide how to schedule all the jobs assigned to it, on its local resources. DS keeps track of popularity of each dataset currently available make data replicating decision. On this architecture, the authors developed and evaluated various replicating and scheduling strategy to study the impact of the two systems. The result showed the importance of data locality in scheduling job.

Follow on, in OptorSim [27], [28], data replication is combined with job scheduling in a two-stage optimization mechanism. Our proposed architecture is the combination and improvement of both above architectures. More recent works by Chakrabati, A. and colleagues [29], [26] or Tang, M et al. [30] improve the older works by integrating the scheduling and replication strategy to improve the scheduling performance.

In contrast to traditional data replication where a complete file is replicated, Chang *et al.* proposed a new data replication approach called fragmented replicas, which only replicates needed partial contents of a file locally to save storage space [31]. However, the problem of fragmented replica updates is to be solved and it is challenging. Also, their assumption that fragmented replicas are contiguous may not always hold in data grids. Further, they developed a job scheduling policy HCS (Hierarchical Cluster Scheduling) and a dynamic data replication strategy HRS (Hierarchical Replication Strategy) to improve the data access efficiencies in a cluster grid [20, 21, 22]. The rationale behind the tow coordinated algorithms is that job scheduling in concert whit data replication can significantly reduce data access time and the amount of inter-cluster-communications.

In [4] proposed replication mechanism. In replication mechanism, each replication server maintains data accesses record. When it is time to replica data, all replication servers send the access record to the central replication manager. The manager will aggregate and create a summarized access record for every unique file identifier (FID). The result is a record NOA which each item NOA(f) indicates the times that a file with unique ID f is accessed on the whole grid system. Once the average number of accesses is calculated, if a replica of a file is accessed more than the average access, then the file needs to be replicated. But they do not use the original number of access in general. Instead, they calculate the amount of data access by taking into account the file size as a parameter when making decision.

In [12], proposed replication strategy, called FIRE. When FIRE needs to make a replication of a file and there are multiple copies of the file in a data grid, FIRE has to decide which site to choose to make the replication. Obviously FIRE prefers to make a replication from a site that will incur the least cost in terms of access time and the site’s job queue cost. The job queue cost of a site is defined as the earliest time that a job can start its execution if it is assigned on the site.
K. Sashi and et al. [2] presents a dynamic replica creation and placement algorithm which can automatically maintain the data by the status of data. Replica creation decides which file to be replicated and how many replications to be created. Replica placement deals with the placement of replicas in appropriate locations. Two factors are considered for the placement of replicas, the response time and the bandwidth.

Analyzing above works, we proposed a new job scheduling algorithm and data replication policy that reduce job execution time by reducing job data access time. Our new scheduling policy considers the locations of required data and the job queue length of a computing node. It is called ESS (Enhancement of Scheduling Strategy). We develop a replication strategy, called ERS (Enhancement of Replication Strategy). It takes into account bandwidth as an important factor for replica selection and replica placement. It also increases the chances of accessing data at a nearby node and uses one mechanism for prevention of full storage.

3. PROPOSED STRATEGY

Since the grid system is distributed, the performance of networks plays an important role in job scheduling and dispatching. In this paper, we propose tow strategy for data replication and enhancement job scheduling by considering the hierarchical network structure. In figure 1, a simple hierarchical form of a grid system, called cluster grid, is shown. A cluster represents an organization unit which is a group of sites that are geographically close.

The size of the data used on Data Grid is from terabytes to petabytes. Scheduling jobs to suitable grid sites is necessary because data movement between different grid sites is time consuming. The scheduling decisions should be based on the appropriate resources a grid site has. Other factors to be considered include CPU workload, features of computational capability, location of data and network load. If a job is scheduled to a site where the required data are present, the job can process data in this site without any transmission delay for getting data from a remote site.
Data replication is another important optimization step to manage large data by replicating data in geographically distributed data stores. Previous replication strategies show that replicating data can offer high data availability and low bandwidth consumption. When users' jobs access a large amount of data from remote sites, dynamic replica optimizer running in the site tries to store replicas on local storage for future possible repeated requests. If most data resides on the same site where they are needed, the frequency of remote data access is going to decrease. This can reduce job execution time and increase the robustness of grid application. Inter-cluster communications also can be avoided, if data within the same cluster are always accessed first [20,2].

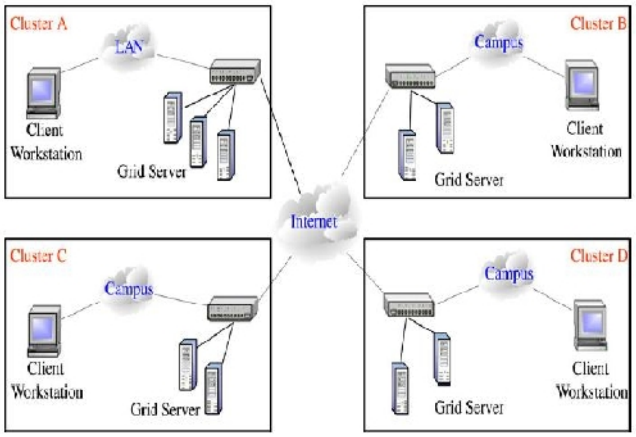


Figure 1. A cluster grid architecture.

3.1. System Architecture

The Data Grid architecture supporting data replication and job scheduling, called cluster grid, is shown in Figure 2. A cluster represents 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 a cluster grid, 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 system 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.

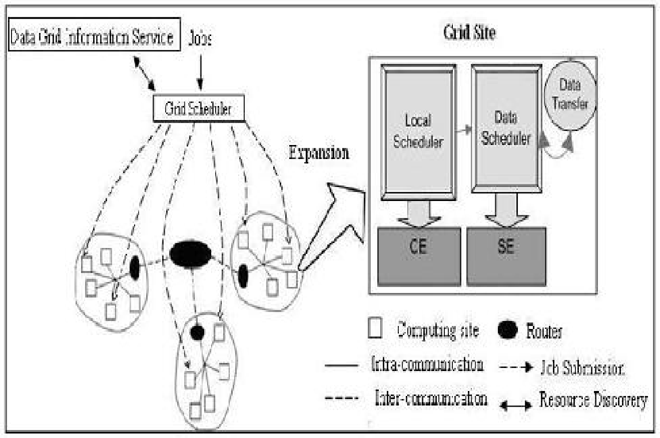


Figure 2. Data Grid Architecture.

Data Grid Information Service providing resource registration services and keeping track of a list of resources available in the Data Grid. The Grid Scheduler can query this for resource contact, configuration, and status information; Resource discovery identify resources that can be used with their capability through Data Grid Information Service. In this structure we apply distributed dynamic data replication infrastructure. Replica Manager at each site manages the data movement between sites and we add Data Scheduler into this structure as part of Replica Manager. When a job is assigned to LS1, the DS2 will responsible for all the data requests by the LS. This data request is generated as soon as job is scheduled into LS queue.

3.2. Scheduling Strategy

In data-intensive applications, data transfer time is the primary cause of job execution delay, the locations of data required by the job impact the Grid scheduling decision and performance greatly. ESS (Enhancement of scheduling strategy) considers the locations of required data at cluster level and site level in job scheduling decision. The algorithm determines the best cluster and site respectively and then submit job to LS. A best cluster (and site) is a cluster that holds most of the requested files (from size point of view).

T This will significantly reduce total transfer time, and reduce the number of inter-communications. We assume that $T_{S,C}$ is total size of the requested files available in site S and cluster C. Also, we assume T_c is total size of the requested files available in cluster C and the needed data to execute job j are represented as: Nj= {RF1, RF2, RFn} and presents size of file RFi. Additionally we define RLi that presents the relative load of site i (based on number of jobs at site i and computing capacity of site i). Q is for all available RF_i in site S.

$$T_{S,C} = \sum_{Q} |RF_i| \tag{1}$$

$$T_c = \sum_{S=1}^{N} T_{S,C} \qquad (2)$$

$$\text{RLi= SJi / Ci} \qquad (3)$$

Where N is number of sites in cluster C, Ci is computing capacity (in MIPS) of site i and SJ_i(SizeofJobsi) is the lengths of queued jobs (in MIPS) on site i. The ESS can be summarized as follow:

1. Compute CC for each cluster in data grid from (2).
2. Select the best cluster $C_{\max} = \underset{i=1}{\overset{q}{MAX}}\, C_i$; where q is the number of cluster (i.e. cluster with largest available requested data files).
3. For each site in $C_{\max}$ compute $S_{S,\max}$ from (1).
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 largest available requested data files from size point of view in cluster $C_{\max}$).
5. If there are several sites whit most available data in the $C_{\max}$ select the site with minimum relative load from (3).

When a job is assigned to LS, the DS will be responsible for transferring all the data required by the job that aren't exists in local site. The objective of this DS is to transferring the data required by the job to the local site before job execution and it leads to reduction of job execution time. The proposed scheduling algorithm by using the DS and overlapping between queue waiting time and data transferring time, improved the job execution time. Additionally, where there are the same data requirements in several sites, ESS takes into consideration the computing capacity of sites with a view to reducing the queue waiting time.

3.3 Replication Strategy

After a job is scheduled to Sj, the requested data will be transferred to Sj to become replicas. ERS (Enhancement of Replication Strategy) determines which replica will be transferred to Sj and how to handle this new replica, as shown in Figure 3. ERS considers the inter-communication bandwidth as the main factor for replica selection / deletion.

For each needed file for job executing, replica manager controls the existence of the file in local site; if file doesn't exist in the local site ERS 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 the new replica 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 required replica is not in a site)
{
    Search replica in the same cluster;
    If (replica is in the same cluster)
        Create list of candidate replicas in the same cluster
    else {
        Search replica in other clusters
        Create list of candidate replicas in other cluster;
    }
    Select the replica with most available bandwidth in Candidate replica list
    If (there is enough space to Store the new replica)
        {Store it; Exit ;} and the following code is run at the random time:
        {If (available space<10% total space)
            While (available space>20% total space)
                Use LRU replacement algorithm to Delete
        }
    else {
        If (replica is in the same cluster)
            {Exit; // avoid duplication in the same cluster; }
        else {
            While (! enough space)
                Use LRU replacement algorithm to Delete Replica in local storage that duplicate in the same cluster;
            If (enough space) {store it; Exit ;}
            else {
                While (! enough space)
                    Use LRU replacement algorithm to Delete Replica in local storage that duplicate in the other cluster;
                If (enough space) {store it; Exit ;}
            }
        }
    }
}
```

Figure 3. Enhancement of Replication Strategy.

If the file doesn’t exist in the same cluster, then ERS 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 room 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 ERS uses LRU replacement algorithm to delete replica in local storage which is duplicated in the other cluster , till it has enough room for the new replica. To conclude, ERS considers 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.

Additionally, ERS controls site space at random time, if available space is less than 10% of total space then **uses LRU** replacement algorithm to Delete, until available space is more than 20% of total space.

4. EXPERIMENTAL RESULTS

The GridSim toolkit provides a comprehensive facility for simulation of different classes of heterogeneous resources, users, applications, resource brokers, and schedulers. It can be used to simulate application schedulers for single or multiple administrative domains distributed computing systems such as clusters and Grids. Application schedulers in the Grid environment, called resource brokers, perform resource discovery, selection, and aggregation of a diverse set of distributed resources for an individual user [15], [17].

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 [16]. 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 4. A view of the simulation in Gridsim is illustrated in figure 5.

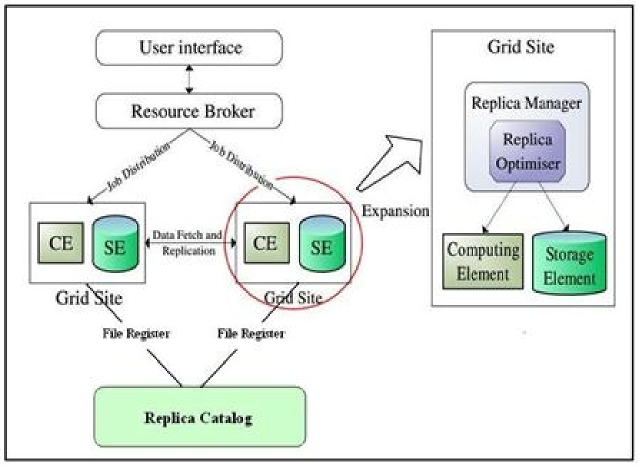
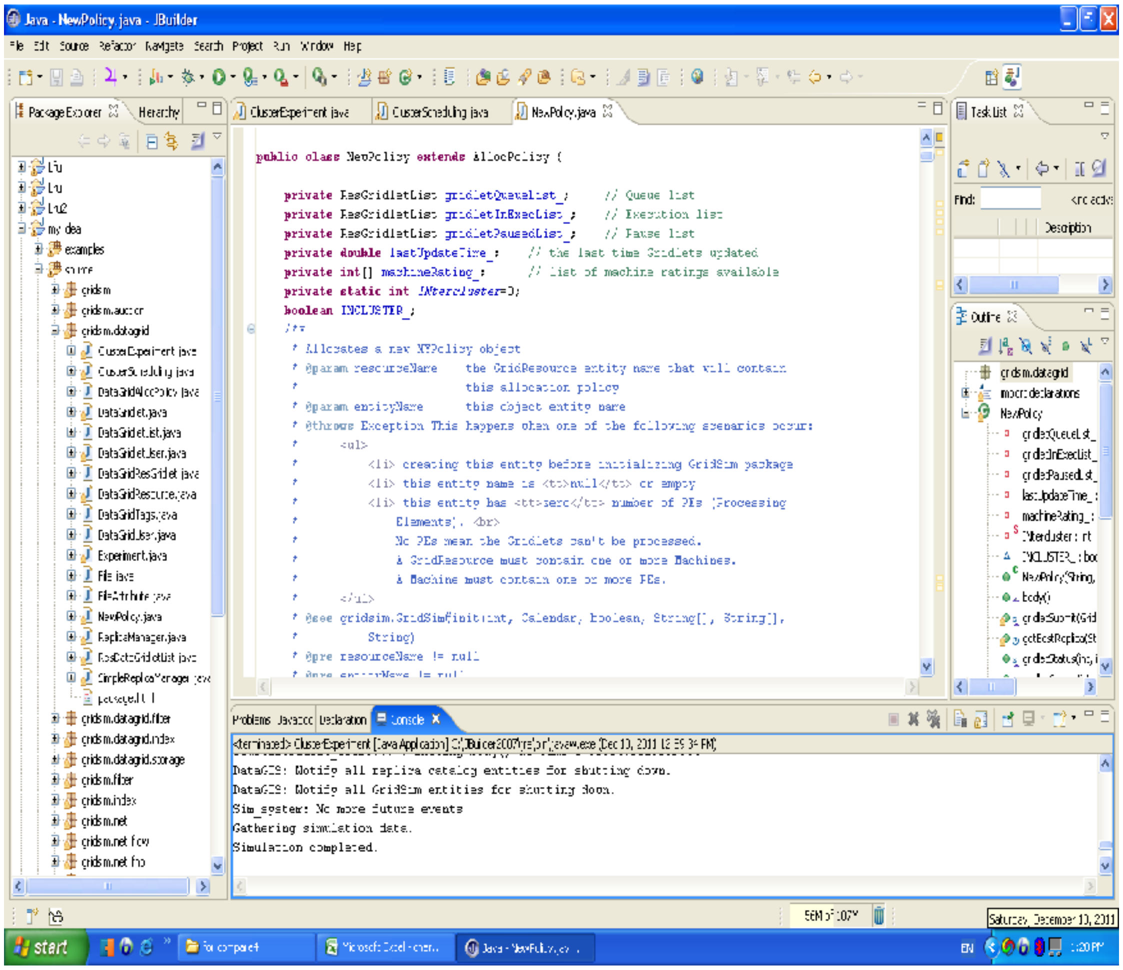


Figure 4. Simulator architecture.


Java - NewPolicy.java - JBuilder
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Hierarchy
ClusterExperiment.java
ClusterScheduling.java
NewPolicy.java
Task List
Outline
Problems Javadoc Declaration Console
start

```
public class NewPolicy extends AllocPolicy {

    private ResGridletList gridletQueueList_;     // Queue list
    private ResGridletList gridletInExecList_;    // Execution list
    private ResGridletList gridletPausedList_;    // Pause list
    private double lastUpdateTime_;    // the last time Gridlets updated
    private int[] machineRating_;      // list of machine ratings available
    private static int INtercluster=0;
    boolean INCLUSTER_;
    /**
     * Allocates a new NYPolicy object
     * @param resourceName    the GridResource entity name that will contain
     *                        this allocation policy
     * @param entityName      this object entity name
     * @throws Exception This happens when one of the following scenarios occur:
     *      <ul>
     *          <li> creating this entity before initializing GridSim package
     *          <li> this entity name is <tt>null</tt> or empty
     *          <li> this entity has <tt>zero</tt> number of PEs (Processing
     *              Elements). <br>
     *              No PEs mean the Gridlets can't be processed.
     *              A GridResource must contain one or more Machines.
     *              A Machine must contain one or more PEs.
     *      </ul>
     * @see gridsim.GridSim#init(int, Calendar, boolean, String[], String[],
     *          String)
     * @pre resourceName != null
```

```
<terminated> ClusterExperiment [Java Application] C:\JBuilder2007\jre\bin\javaw.exe (Dec 10, 2011 12:59:34 PM)
DataGIS: Notify all replica catalog entities for shutting down.
DataGIS: Notify all GridSim entities for shutting down.
Sim_system: No more future events
Gathering simulation data.
Simulation completed.
```

Figure 5. A view of simulation.

Resource Broker: It receives jobs from user and sends them to the best node according to proposed algorithm. **Storage Element (SE):** Storage resource in grid. **Computing Element (CE):** Computing resource in grid. **Replica Catalogue (RC):** 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. **Replica Manager:** It controls data transferring in each node and provide a mechanism for accessing the Replica Catalogue. **Replica Optimizer:** It has replica algorithm and control file replication according to proposed algorithm. Based on the scheduling algorithm the broker sends jobs to a node. 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, 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

ERS will be compared with LRU (Least Recently Used), LFU (Least Frequency Used), BHR (Bandwidth Hierarchy based Replication), HRS(Hierarchical Replication Strategy) and ERS(Enhancement of Replication Strategy). The LRU algorithm always replicates and then deletes those files that have been used least recently and LFU algorithm always replicates and then deletes those files that have been used least frequently used and both algorithms search all sites to find the best replica or to delete replica on the basis of LFU or LRU method. Figure 6 shows the Average job time based on changing number of jobs for 5 algorithms. ERS replication strategy uses the concept of “network locality” as BHR [5]. The difference between ERS and BHR is that required replica within the same cluster is always the top priority used in ERS, 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 ERS will avoid inter-cluster-communications and be stable in hierarchical network architecture with variable bandwidth. The difference between ERS and HRS is that HRS uses LFU replacement algorithm to delete, whereas ERS uses LRU replacement algorithm to delete. Our method takes benefit from network level locality of BHR and HRS. Thus, total job execution time is about 10% and 30% faster using ERS optimizer than HRS and BHR, respectively.

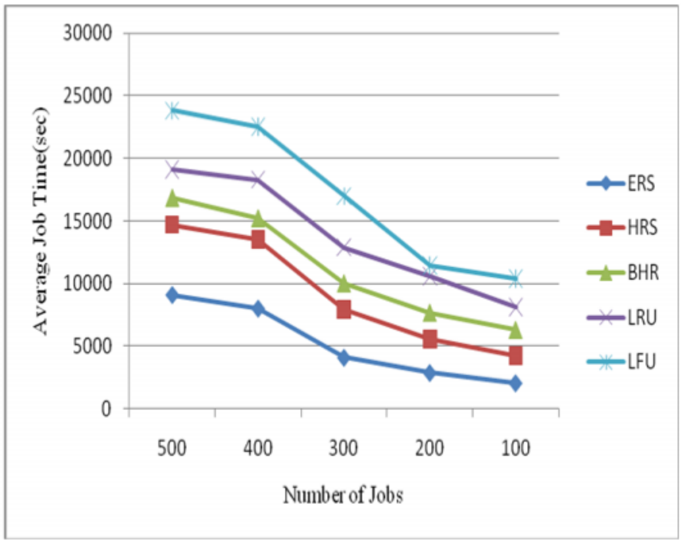


Figure 6. Average job Time based on varying number of jobs.

The experimental results of average job execution time are presented in Figure 6. The average job execution time for one job is obtained through dividing the overall job experimental time by the number of jobs. As mentioned above, the job execution time is the file transmission time plus job processing time. Since the file transmission time is the most important factor to influence the job execution time for data intensive jobs in data grids, ESS with ERS can reduce the file transmission time effectively by virtue of valid scheduling and proper data replication, as can be seen from the experiments. Figure 7, shows the experimental results of average access time based on varying number of jobs for 3 algorithms, BHR, HRS, ERS.

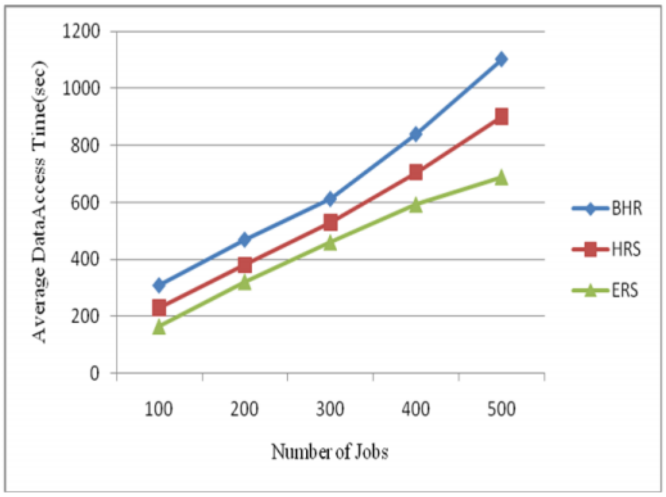


Figure 7. Average access time based on varying number of jobs.

For replication strategies, LRU and LFU show similar performance in Figure 8. The same results are obtained in [25]. We implement BHR replication strategy into GridSim. Total job execution time is about 30% faster using BHR optimizer than LRU and LFU. Our method takes benefit from network level locality of BHR and HRS and we simplify those replica replacement model. Thus, ERS successfully accelerates the total execution time up to 25% whether in DPRL or ESS.

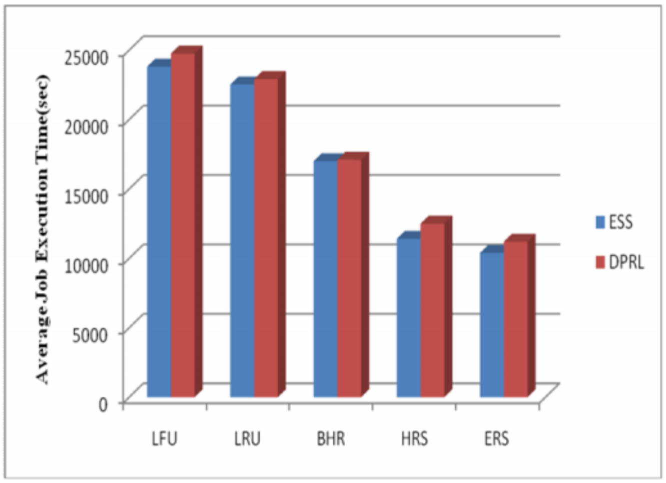


Figure 8. Average job Execution Time for various job scheduling and replication algorithms.

In the ESS the job execution time is the Max {file transmission time, queue time} plus job processing time. ESS will be compared with DPRL. 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 9 shows the Average job time based on changing number of jobs for 2 algorithms. Since ESS 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, ESS with ERS can reduce the file transmission time effectively by virtue of valid scheduling and proper data replication.

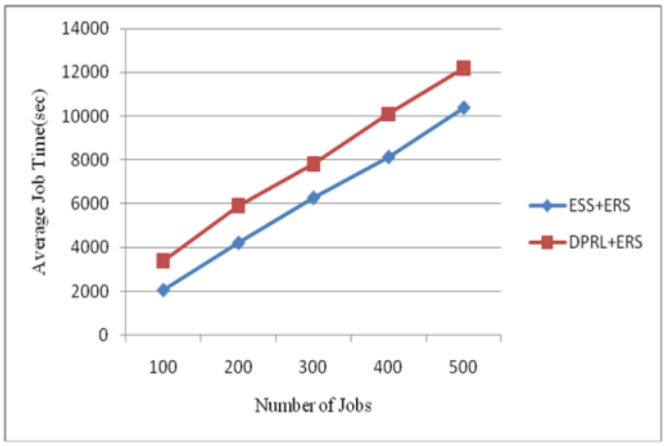


Figure 9. Average job Time based on varying number of jobs.

The average number of inter-communications for a job execution is illustrated in Figure 10. By selecting the best cluster and best site, ESS with ERS can decrease the number of inter-communications effectively. Overall the simulation results with Gridsim show better performance (over 30%) comparing to current algorithms.

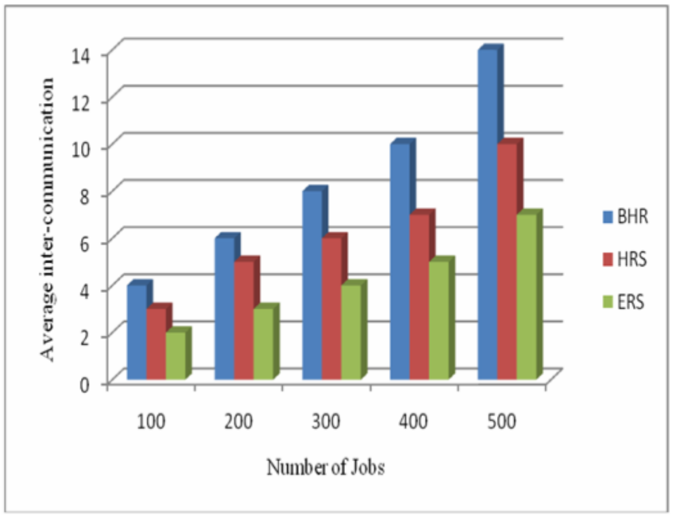


Figure 10. Average number of inter-communications

Figures 11 shows the average job time for 500 jobs. We compare ERS, HRS and BHR algorithms for varying inter-communication bandwidth.

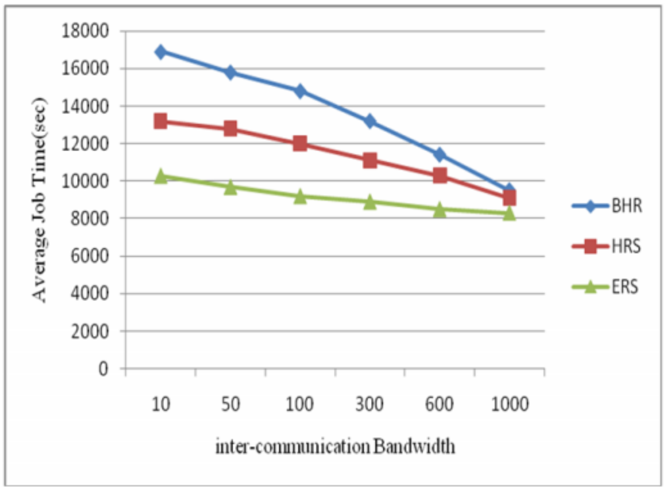


Figure 11. Average Jobs Time with varying inter- communication bandwidth for 500 jobs.

The result of average data access time based on varying inter- communication bandwidth for 3 algorithms are shown in figure 12. ERS show performance improvement and low data access time as compared with other both algorithms.

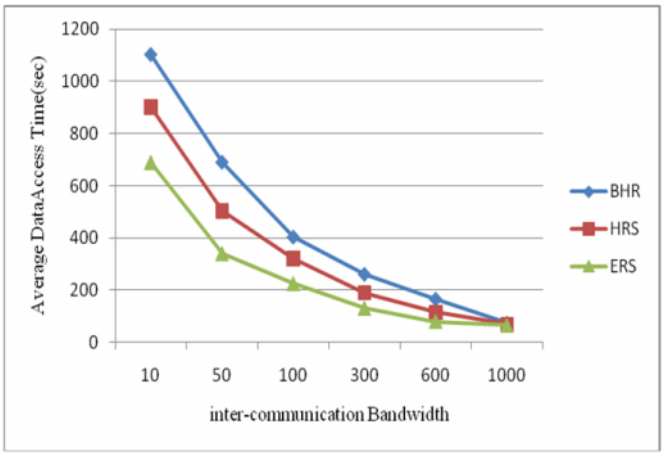


Figure 12. Average data access time based on varying inter-communication bandwidth.

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

6. CONCLUSION AND FUTURE WORK

We study and evaluate the performance of various replica strategies and different algorithm combinations. In this paper a two layered 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 (ESS) that considers not only computational capability, job type and data location but also it considers cluster information in job placement decision. The simulation results show, first of all, that ESS and ERS both get better performances than other scheduling policy and replica strategies. Second, we can achieve particularly good performance with ESS where jobs are always scheduled to cluster with most of the needed data, and a separate ERS process at each site for replication management. Experimental data shows ESS scheduling with ERS replica strategy outperforms others combinations in total job execution time and data access time.

ESS and ERS can be applied to and embedded to any grid systems. For example, the Taiwan Ecogrid project [32], [23] has deployed many sensors in several ecological areas in Taiwan to gather environmental data and real time monitoring video for ecological analysis. The data must be replicated and distributed over various areas and grid sites for processing. It is obvious that the size of ecological data is quite large. Ecology research jobs processing large amounts of environmental data consume considerable network bandwidth and computing resources if without an appropriate scheduling algorithm and data replication strategy. ESS with ERS strategy could be applied to such an ecological grid computing environment to improve the system performance. In this paper, we have proposed our improvement over current Grid scheduling and replication problem. In replication issue, we have proposed a new approach to replication problem in Data Grid. The simulation result showed that our replication placement strategy overcomes the random placement strategy. Also, the dynamic replicating algorithm makes an improvement and can be used with GridSim's replication optimization. The result is quite promising. In the future work, we will do more simulation test and improve the replication strategy. Meanwhile, the scheduling component needs to be completed for integrating with replication mechanism to perform a whole system simulation.

REFERENCES

[1] I. Foster, C. Kesselman, "The Grid Blueprint for a new computing infrastructure," Morgan Kaufman, 2004.

[2] K. Sashi, A.S. Thanamani, "Dynamic Replica Management for Data Grid," IACSIT International Journal of Engineering and Technology, Vol.2, No.4, pp. 329-333, August 2010.

[3] A. Horri and et al, "A Hierarchical Scheduling and Replication Strategy," IJCSNS International Journal of Computer Science and Network Security, Vol.8, No.8, pp.30-35, August 2008.

[4] N.N. Dang, S.B. Lim, "Combination of Replication and Scheduling in Data Grids," IJCSNS International Journal of Computer Science and Network Security, Vol.7, No.3, pp.304-308, March 2007.

[5] D.T. Nukarapu and et al, "Data Replication in Data Intensive Scientific Applications with Performance Guarantee," IEEE Transactions on Parallel and Distributed Systems, Vol.22, 2011.

[6] P.K. Suri, M. Singh, "DR2: A Two-Stage Dynamic Replication Strategy for Data Grid," International Journal of Recent Trends in Engineering, Vol.2, No.4, November 2009.

[7] J. Jiang and et al, "Scheduling Algorithm with Potential Behaviors," Journal of Computers, Vol.3, No.12, pp. 51-59, December 2008.

[8] H. Lamehamedi and et al, "Simulation of Dynamic Data Replication Strategies in Data Grids," Proc. 12th Heterogeneous Computing Workshop (HCW2003) Nice, France, April 2003. IEEE Computer Science Press.

[9] M. Tang and et al, "The Impact of Data Replication on Job Scheduling Performance in the Data Grid," Future Generation Computer, Vol.22, pp. 254-268, 2006.

[10] M. Tang and et al, "Dynamic Replication Algorithms for the Multi-Tier Data Grid," Future Generation Computer, Vol.21, pp. 775-790, 2005.

[11] K. Ranganathan, I. Foster, "Computation Scheduling and Data Replication Algorithms for Data Grids," National Science Foundation's GriPhyN Project, 2003.

[12] A.R. Abdurrab, T. Xie, "FIRE: A File Reunion Based Data Replication Strategy for Data Grids," 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, pp. 215-223, 2010.

[13] S.M. Park and et al, "Dynamic Data Grid Replication Strategy based on Internet Hierarchy," International Mobile Telecommunications 2000 R&D Project, 2003.

[14] A. Chakrabarti, S. Sengupta, "Scalable and Distributed Mechanisms for Integrated Scheduling and Replication in Data Grids," ICDCN, pp. 227-238, 2008.

[15] R. Buyya, M. Murshed, "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing," Concurency and Computation: Practice and Experience, Vol.14, pp. 1175–1220, 2002.

[16] M. Murshed, R. Buyya,"Using GridSim Toolkit for Supercharging Grid Computing Education," 2002.

[17] A. Sulistio and et al, "Visual Modeler for Grid Modeling and Simulation (GridSim) Toolkit," ICCS, pp. 1123-1132, 2003.

[18] H. Stockinger and et al, "File and Object Replication in Data Grids," National Science Foundation, 2001.

[19] A. Takefusa and et al, "Performance Analysis of Scheduling and Replication Algorithms on Grid Datafarm Architecture for High-Energy Physics Applications," MEXT, 2002.

[20] 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, pp. 846-860, August 2007.

[21] S. Abdi, S. Mohamadi, "The Impact of Data Replication on Job Scheduling Performance in Hierarchical Data Grid," International journal on applications of graph theory in wireless ad hoc networks and sensor networks (GRAPH-HOC) Vol.2, No.3, September 2010.

[22] S. Abdi, S. Mohamadi, "Two Level Job Scheduling and Data Replication in Data Grid," International Journal of Grid Computing & Applications (IJGCA) Vol.1, No.1, September 2010.
[23] The Data Grid Project. http://www.eu-datagrid.org.
[24] I. Foster, "The Grid: A new infrastructure for 21st century science," Physics Today 55, pp. 42-47, 2002.
[25] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The data grid: Towards an architecture for distributed management and analysis of large scientific datasets," Journal of Network and Computer Applications 23, pp. 187–200, 2000.
[26] K. Ranganathan, I. Foster, "Simulation Studies of Computation and Data scheduling Algorithms for Data Grids," Journal of Grid Computing, vol.1, Issue 1, pp. 53-62, 2003.
[27] W.H. Bell and et al, "Evaluation of an Economy-Based File Replication Strategy for a Data Grid. In International Workshop on Agent based Cluster and Grid Computing at CCGrid 2003," Tokyo, Japan, May 2003. IEEE Computer Society Press.
[28] D.G. Cameron and et al, "Evaluating Scheduling and Replica Optimization Strategies in OptorSim," In 4th International Workshop on Grid Computing (Grid2003), Phoenix, Arizona, November 17, 2003. IEEE Computer Society Press
[29] H. Lamehamedi and et al, "Simulation of Dynamic Data Replication Strategies in Data Grids," In Proc. of 12th Heterogeneous Computing Workshop (HCW2003), Nice, France, Apr 2003. IEEE-CS Press.
[30] T.M. Lee and et al, "The impact of data replication on job scheduling performance in the Data Grid," Future Generation Computing System 22, pp. 254-268, February 3, 2006.
[31] R.S. chang, P.H. Chen, "Complete and fragmented replica selection and retrieval in data grids," Future Generation Computer Systems, Vol.23, Issue 4, pp. 536-546, may 2007.
[32] Taiwan Ecogrid project. http://ecogrid.nchc.org.tw.