2009 First International Conference on Computational Intelligence, Communication Systems and Networks

Agent Based Replica Placement in a Data Grid Environment

Ms. Shaik Naseera
Department of Computer Science & Engineering
Sri Venkateswara University
Tirupati, India
naseerakareem@gmail.com

Dr. K. V. Madhu Murthy
Professor, Department of CSE
Sri Venkateswara University
Tirupati, India
kvmmurthy@yahoo.com

Abstract— **In a data grid, large quantities of data files are produced and data replication is applied to reduce data access time. Determining when and where to replicate data in order to meet performance goals in grid systems with many users and files, dynamic network and resource characteristics and changing user behavior is difficult. Therefore efficiency and fast access to replicated data are influenced by the location of the resource holding the replica. In this paper, we present an agent based replica placement algorithm to determine the candidate site for the placement of replica. An agent is deployed at each site holding the master copies of the shared data files. To create a replica, each agent prioritizes the resources in the grid based on the resource configuration, bandwidth in the network and the demand for the replica at their sites and then creates a replica at suitable resource locations. We have carried out the simulation using GridSim Toolkit-4.0 [1] for EU Data Grid Testbed1 [14]. The simulation results show that the aggregated data transfer time and the execution time for jobs at various resources is less for agent based replica placement.**

Keywords- Data Grid, Data Replication, Candidate Site, Resource Location, GridSim Toolkit-4.0.

I. INTRODUCTION

A grid environment offers a large number of similar or equivalent resources that the grid users can select and use them for the execution of their applications. These resources may provide same functionality but offers different QoS measures. The basic performance measure is the total time required for completing the execution of the given application.

Data Grids provide geographically distributed storage resources for complex computational problems that require the evaluation and management of large amounts of data [3,4,5]. Experiments like high-energy physics, earthquake engineering, climate modeling generate massive amount of datasets, which need to be shared and analyzed. A community of hundreds or thousands of researchers distributed worldwide must share these datasets.

An important technique that speeds up the data access in data grid systems is replication of data in multiple locations so that a user can access it from a site in his vicinity. It is shown that data replication not only reduces access cost but also increases data availability in many applications [6,7,8].

As the data is voluminous, the cost of maintaining local copies of data at each site is also expensive. Although a substantial amount of work has been done on data replication in grid environments, most of it has focused on the mechanism for creating/deleting replicas [9,10,11]. We believe that to obtain the maximum benefit from replication, strategic selection of the replica location is also important. Thus in this paper we address the problem of replica location selection process in a data grid systems. The study of our replica placement algorithm is carried out using a model of the EU-Data Grid Testbed [14]. In our approach, we propose an agent based replica placement algorithm for making a decision to select a candidate site for replica placement. The agent in this approach is autonomous, self-contained software capable of making independent decisions.

Our replica placement strategy considers two important issues. First issue in choosing a replica location is to place a replica at sites that optimize the aggregated response time. This issue can be addressed by placing replica in a proper location so that the time taken for obtaining all the files required by the job is minimized. Second issue in choosing a replica location is to place a replica at sites that optimize the total execution time of the jobs executed in the grid.

Response time is calculated by multiplying the number of requests at site i with the transmission time between the nearest replication site to the requester. The sum of the response times for all sites constitutes the aggregated response time.

The rest of the paper is organized as follows. Section 2 reviews related work. In section 3, we describe agent-based replica placement approach. In section 4, we described the simulation framework. The experimental results are presented in section 5. Section 6 provides conclusions.

II. RELATED WORK

Kavitha and Foster [2] discuss various replication strategies for hierarchical data grid architecture. They proposed six different replication strategies such as no replication, best client, cascading, plain catching, caching

978-0-7695-3743-6/09 $25.00 © 2009 IEEE
DOI 10.1109/CICSYN.2009.77

plus cascading and fast spread for three different kinds of access patterns.

Rahman et. al. [13] proposed a static replica placement algorithm that places replicas to sites by optimizing average response time and a dynamic replica maintenance algorithm that re-allocates replicas to new sites if performance reduces over last k time periods. It is also demonstrated in [17] that the data replication reduces access cost and improves the data availability in many applications.

Lamehamedi et. al. [6] have proposed replica management service that offer high availability, low bandwidth consumption, increases fault tolerance and improved scalability of the overall system. They made a replication decision based on the cost model that considers the data access cost and performance gains of creating replicas.

Rosa Badia et. al. proposed a reliability and trust based workflow job mapping on the grid by considering application performance, computation and data transfer capacity to predict a reliable resource [18]. Haddad and Slimani [19] proposed an economic model for the problem of database placement in grid architecture. They deployed two agents to drive the economic model for the database placement.

Globus toolkit middleware provides a replica management service for managing multiple copies of shared data sets [21]. It uses a replica catalog to register replicas and provide mappings between logical names to the storage locations of the replicas.

Several research efforts [20,12] consider user request for replica placement and ignore network latencies. However, network bandwidth plays a vital role in large file transfers. Substantial transfer time is saved if we place file replicas at neighboring sites with limited bandwidth but high request rates.

III. AGENT-BASED SYSTEM MODEL

A software agent can be defined as a software entity which functions continuously and autonomously in a particular environment and which is able to carry out activities in a flexible and intelligent manner that is responsible for changes in the environment. The main objective of an agent is to select a candidate site for the placement of a replica that reduces the access cost, network traffic and aggregated response time for the applications. To select a candidate site for a replica, an agent is deployed at each site that holds the master copies of the files for which replicas are to be created.

An agent at each site makes the decision based on resource factors that influence the data transmission time between the sites. The factors include baud-rate between the sites, CPU Rating, CPU Load, Site Storage Capacity and Local Demand of the replicas at each site. The agent uses a Multi-Dimensional Ranking (MDR) function to evaluate the resource properties and grade with an appropriate rank. The agent preferences are represented by a set of factor weightings, which allow resource rank to be tailored to the current resource characteristics.

In this paper we conducted the simulation for EU- Data Grid Tesebed1 [14] in which master files are initially placed at CERN storage. The grid configuration is shown in Table2.

A. Multi dimensional ranking function (MDR)

Let j be a site holding master copies of the data files. An agent A_j at Site j assigns a rank R_i to a resource i with respect to a particular goal G, based on the parameters namely baud-rate, load, local demand, CPU rating and storage capacity.

1) ***Baud-rate (br_i):*** *This parameter represents the configuration of the resource communication channel in the grid network. It influences the transmission time for the file from the replica location to the requesting site.*

2) ***Load (l_i):*** *This parameter represents the peak load of the CPU in the resource. The CPU Load is generally measured in terms of number of processes waiting in the queue. The CPU load varies at different intervals of time. A highly loaded resource responds slowly compared to the lightly loaded resource.*

3) ***Local Demand (ld_i):*** *This parameter represents the list of files needed for the jobs execution. When all the required files are available on the local storage, the job is scheduled for execution. It is recommended to place a replica at a site that has greater number of access to the replicated files.*

4) ***CPU Rating (r_i):*** *The processing capability of a resource's CPU is modeled in terms of MIPS (Million Instructions per Second). Higher CPU rating substantially reduces the average execution time of the submitted jobs.*

5) ***Storage capacity (s_i):*** *An individual storage is responsible for storing, retrieving and deleting files. For data intensive applications, larger storage resources are preferred as they involve large volumes of data sets.*

We define the Rank R_i as shown in eq (1).

$$R_i = br_i \times a_1 + l_i \times a_2 + ld_i \times a_3 + r_i \times a_4 + s_i \times a_5 \quad (1)$$

The factor weightings a_i defined in R_i must be chosen such that

$$a_1 + a_2 + a_3 + a_4 + a_5 = 1 \quad (2)$$

A resource with highest rank ensures better aggregate response time compared to resource with lower rank. We have conducted experiments on varying range of factor weighting on the resource properties. A range of factor weightings is investigated as shown in Table 1.

TABLE I. FACTOR WEIGHTINGS

Factor ID	br_i	l_i	ld_i	r_i	s_i
Factor1	0.2	0.2	0.2	0.2	0.2
Factor2	0.3	0.2	0.2	0.3	0.0
Factor3	0.0	0.35	0.35	0.3	0.0

The agent's behavior for these factor weightings is discussed in the section 5 under 3 different cases.

IV. SIMULATION FRAMEWORK

To evaluate our approach we use a simulation package called GridSim 4.1. GridSim toolkit is a data grid simulator. It provides the ability to define resources with heterogeneous storage components and the flexibility to implement various data management strategies like creation, deletion and replication of files in a grid environment [1].

In this paper we extended the GridSim toolkit to incorporate the functionality of agent for decision-making process for the selection of candidate site.

A. *Architecture*

The simulation design consists of number of data resources each consisting of one or more processing elements (PE) with one or more storage elements. The PE's provide computational capability and storage elements serves as data storage resource for submitted jobs. A replica manager handles and manages incoming requests about data sets in a resource for one or more storage elements. It also performs registration of files stored in the resource to a designated replica catalog (RC). The function of a RC is to store the metadata about files and to provide mapping between filename and its physical location.

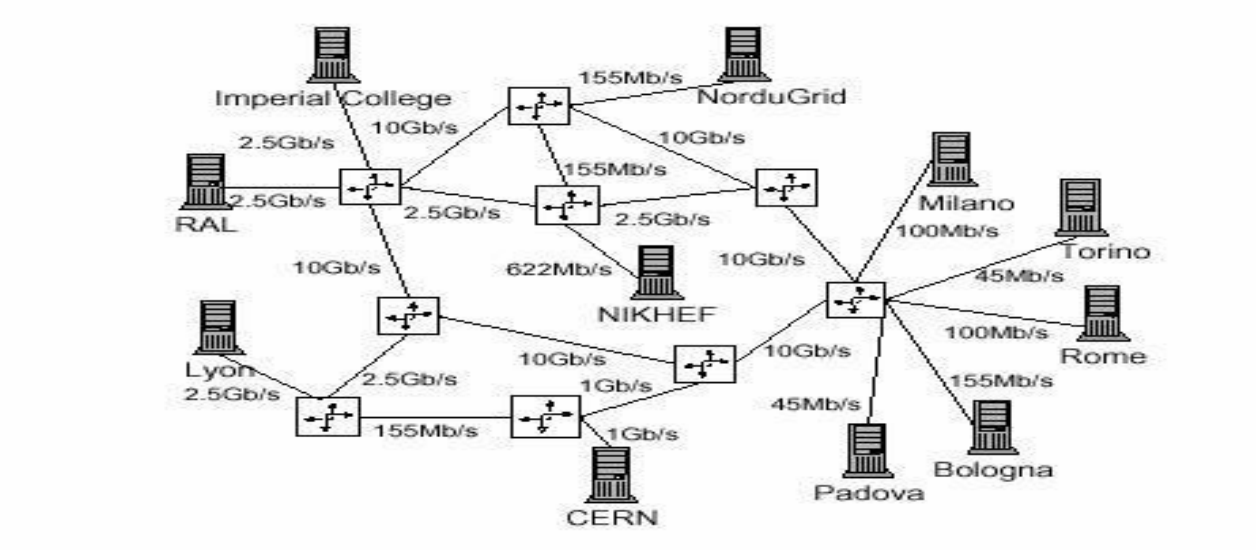

Figure 1. The Simulated Topology of EU DataGrid TestBed1

TABLE II. RESOURCE SPECIFICATION

Resource ID	Resource Name (location)	Storage (TB)	#Nodes	CPU Rating	Policy	Load	Local Demand	#User
Res_0	RAL(UK)	2.75	41	49000	Space-Shared	0.82	50	24
Res_1	Imperial College(UK)	1.80	52	62000	Space-Shared	0.87	40	32
Res_2	NorduGrid(Norway)	1.00	17	20000	Space-Shared	0.69	30	8
Res_3	NIKHEF(Netherlands)	0.50	18	21000	Space-Shared	0.02	25	16
Res_4	Milano(Italy)	0.35	5	7000	Space-Shared	0.34	25	8
Res_5	Torino(Italy)	0.10	2	3000	Time-Shared	0.84	50	4
Res_6	Rome(Italy)	0.25	5	6000	Space-Shared	0.36	45	8
Res_7	Bologna(Italy)	5.00	67	80000	Space-Shared	0.34	30	24
Res_8	Padova(Italy)	0.05	1	1000	Time-Shared	0.24	25	4
Res_9	CERN(Switzerland)	2.50	59	70000	Space-Shared	0.42	45	48
Res_10	Lyon(france)	1.35	12	14000	Space-Shared	0.62	35	24

A data-intensive job in a GridSim is termed as a DataGridlet. Each DataGridlet has a certain execution size in MI (Million Instructions) and require access to a set of files which may be located at different locations.

After receiving a DataGridlet, the Replica Manager (RM) at each resource checks a list of required files for executing the job. If all the files are currently available on a resource local storage, the DataGridlet is sent to a resource's scheduler for execution. Otherwise RM sends a request for obtaining the needed files from other resources. When all the requested files have been transferred and stored on the local storage, then the DataGridlet is executed by the scheduler.

B. Grid Configuration

The study of our agent based data replication algorithm is carried out based on an EU Data Grid Testbed1 [14]. The resources and their associated network geometry is as shown in Fig 1. Initially all the master files are placed on the CERN (European Organization for Nuclear Research) storage. A master file is an original instance of the file.

Table 2 summarizes all the resource relevant information. The resource's PE rating is modeled in the form of MIPS. We conducted this experiment for 200 files. The average file size is 1GB and the file size follows a power law distribution [15].

We have created 100 types of data intensive jobs. Each job requires 10 to 60 files to be executed. The required files for the DataGridlets are chosen with Zipf-like distribution [16].

The experiment is conducted with 200 DataGridlets with the approximate size of 84000KMI ± 30%. Each resource is provided with a user to submit their jobs approximately for every 1 minute. Each resource, peak load is chosen as specified in the Table 2.

Some parameters are identical for all network links i.e., the Maximum Transmission Units (MTU) is 1500 bytes and the latency of links is 100msec. The RM of each resource follows least-frequently used replica strategy to delete a replica when the storage capacity is full. However, the master files at CERN can not be deleted or modified during the simulation.

V. EXPERIMENTAL RESULTS

Each job is submitted at every approximately 1min interval. Initially all the master files are placed at CERN (Res_9) storage and agent is initially located at CERN. The agent at CERN, assign the ranking to the resources based on the ranking function MDR using different factor weightings as specified in the Table 1.

A. Execution time test

The simulation is carried for 10 DataGridlets each requires 45-60 files for their execution. Fig. 2 depicts the comparison of aggregated execution time of DataGridlets for replica placement at different candidate sites. The agent's recommendation for different factor weightings is described as in the following cases.

*1) **Case I:** As per the factor1 weightings, Res_0 is graded with a best rank by the agent and is selected as a candidate site for the placement of replica.*

*2) **Case II:** As per the factor2 weightings, Res_1 is graded with a best rank by the agent and hence Res_1 is selected as a candidate site for the placement of replica.*

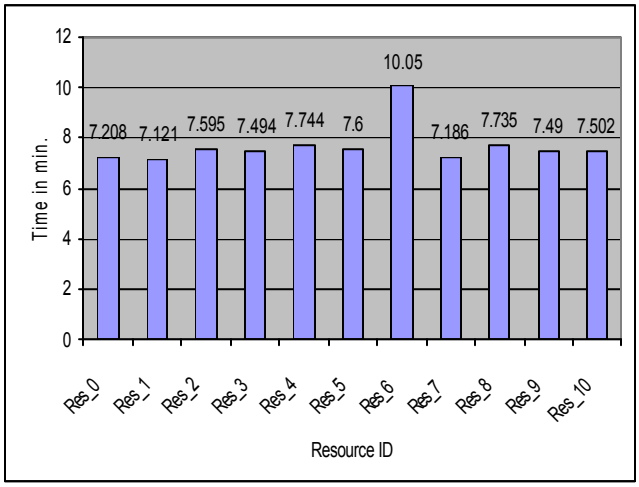

Figure 2. Agrregated Execution Time for Diffrerent Candidate Sites

*3) **Case III:** As per the factor3 weightings, the agent grade Res_7 with a best rank and Res_7 is selected as a candidate site for replica placement.*

The Fig. 2 shows that the replica placement at Res_1 offers lowest aggregated execution time compared to replica placement at other candidate sites. Therefore it is demonstrated that the application which are concerned about the aggregated execution time of the DataGridlets are recommended to follow factor2 weightings by the agent for the selection of candidate site for replica placement.

B. Data Availability Test

The availability of data is measured in terms of how much time a resource spend for obtaining the necessary files for its DataGridlet to execute. This measure tells us how close the resource is to the required data. Fig. 3 depicts the comparison of aggregated data transfer time of DataGridlets for replica placement at different candidate sites.

It is demonstrated in Fig. 3 that Replica placement at candidate site Res_7 offers lowest aggregated data transfer time compared to other candidate sites for replica placement. Lower data transmission time reduces the network congestion. Therefore, applications, which are concerned about the file transmission time, are recommended to follow factor3 weightings by agent for decision making to select a candidate site.

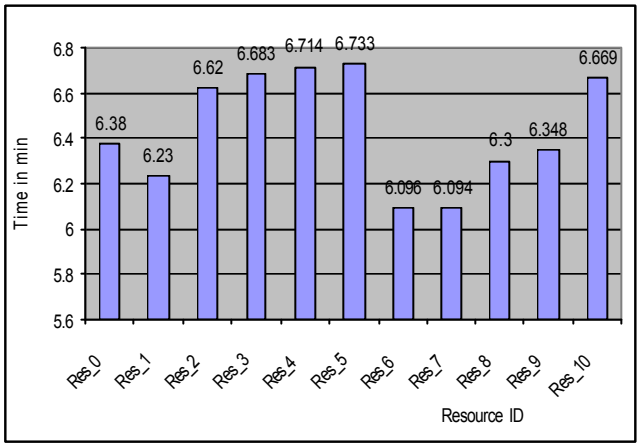

Figure 3. Aggregated Data Transfer Time for Different Candidate Sites

VI. CONCLUSIONS

We proposed an agent-based replica placement algorithm to determine the candidate site for replica placement. We observe that the agent based candidate site selection using factor2 weightings offer lowest aggregated response time and agent based candidate site selection using factor3 weightings offer lowest aggregated data transfer time. The simulation is done using GridSim toolkit 4.1. The agent functionality is incorporated by extending the GridSim toolkit and validated our model for EU Data Grid testbed1.

REFERENCES

[1] A. Sulistio, U. Cibej, B. Robic and R. Buyya, "A Toolkit for Modelling and Simulation of Data Grids with Integration of Data Storage, Replication and Analysis", International journal of Concurrency and Computation-Practice and Experience, 2008, vol 20, pp. 1591-1609.

[2] R. Kavitha and I. Foster, "Design and Evaluation of Replication Strategies for a High Performance Data Grid", Proceedings of the International Conference on Computing in High Energy and Nuclear Physics, Beijing, September 2001.

[3] A. Chervenak, I. Foster, C.Kesselman, C. Salisbury, and S.Tuecke, "The data grid : Towards an architecture for the distributed management and analysis of large scientific datasets", Journal of Network and Computer Applications, October 2000, pp. 187-200.

[4] W. Hoschek, F.J.Janez, A.Samr, H. Stockinger and K. Stockinger, "Data management in an international data grid project", Proceedings of the First IEEE/ACM International Workshop on Grid Computing, Bangalore, India, December 2000, pp. 77-90.

[5] R. Moore, C. Baru, R. Marciano, A. Rajasekar and M. Wan, I. Foster and C. Kessselman edited, "The Grid: Blueprint for a Future Computing Infrastructure", chapter Data intensive computing. Morgan Kaufmann Publishers, 1999.

[6] H. Lamchamedi, B. Szymanski, Z. Shentu and E. Deelman, "Data replication strategies in grid environments", In proceedings of 5th International Conference on Algorithms and Architecture for Parallel Processing Beijing, China, October 23-25, 2002, pp 378-383.

[7] A. Chervenak, R. Schuler, C.Kesselman, S.Koranda and B. Moc, "Wide area data replication for scientific collaborations", In proceedings of the 6th International Workshop on grid Computing, Seattle, Washington, USA, November 2005.

[8] K. Kalpakis, K.Dasgupta nad O. Wolfaon, "Optimal Placement of replicas in trees with read, write and storage costs", IEEE Trans. Parallel Distributed Systems, 12(6), 2001, pp. 628-637.

[9] W. B. David, "Evaluation of an economy-based file replication strategy for a data grid", In International Workshop on Agent Based Cluster and Grid Computing, Tokyo, Japan, Pages 120-126, 2003.

[10] W. B. David, D.G. Capozza, A.P.Millar, K. Stocklinger and F Zini, "Simulation of dynamic grid replication strategies in OptorSim", In Proceedings of 3rd International IEEE workshop on Grid Computing, Baltimore, MD, USA in November 2002, pp 46-57.

[11] M.M. Deris, Abawajy J.H., and H.M. Suzuri, "An efficient replicated data access approach for large-scale distributed systems", In IEEE International Syposium on Cluster Computing and the Grid, Chicago, Illinois, USA, April 2004.

[12] W. H. Bell, D.G. Cameron, A. Paul Millar, L. Capozza, K. Stockinger, and F. Zini, "OptorSim – A grid simulatior for studying Dynamic Data Replication Strategies", International Journal of High Performance Computing Applications, vol. 17, , Nov 2003, pp. 403 – 416.

[13] R.M. Rahman, K. Baker and R. Alhajj, "Replica Placement Design with Static Optimality and Dynamic Maintainability", Proceedings of the IEEE/ACM International Conference on Cluster Computing and Grid (CCGRID 06), Singapore, May 2006.

[14] The European Data Grid Project. Homepage http://eu-datagrid.web.cern.ch/eu-datagrid,2005.

[15] W. Gong, Y. Liu, V. Misra and D. Towsley, "On the tails of web file size distributions", In Proceedings of 39th Allerton Conference on Communication, Control and Computing, Pacific Grove, California, November 2001.

[16] L. Breslau, P. Cao, L. Fan, G. Phillips and S. Shenkar, "Web catching and zipf-like distributions: Evidence and implications", In INFOCOM (1), 1999, pp 126-134.

[17] S. Naseera, T.Vivekanandan and Dr. K.V. Madhu Murthy, "Trust Based Data Replication Strategy in a Data Grid", In proceedings of 2nd International conference on Information Processing, ICIP, Bangalore, June 2008, pp. 467-473.

[18] A.A. Sedrakian, R.M. Badia, T. Kielmann, A. Merzky, J.M. Perez and R. Sirvent "Reliability and trust based workflows' job mapping on the grid", CoreGRID Technical Report Number TR-0069, January 30, 2007.

[19] C. Haddad and Y. Slimani, "Economic model for replicated database placement in grid", In proceedings of 7th IEEE international symposium on Cluster Computing and the Grid (CCGRID' 07), Brazil, May 14-17, 2007.

[20] R. Kavitha, I. Foster, "Decoupling Computation and Data Scheduling in Distributed Data-Intensive Applications", In Proceedings of IEEE International Symposium on High Performance Distributed Computing, Edinburgh, July 2002.

[21] I. Foster and C.Kesselman. "Globus: A metacomputing infrastructure toolkit", International journal of super computer and high performance computing applications, 1997, pp. 115-128.