

Evaluation of an Economy-Based File Replication Strategy for Malaysian Research and Education Network (MYREN) Data Grid Model

Azizol Abdullah (Corresponding author), Rohaya Latip & Wan Mohd Azraei Wan Mustapha

Department of Communication Technology and Network

Faculty of Computer Science and Information Technology

University Putra Malaysia

43400 UPM Serdang Selangor

Tel: 603-89-466-540 E-mail: azizol@fsktm.upm.edu.my

Received: September 14, 2009

Accepted: September 15, 2011

Published: January 1, 2012

doi:10.5539/cis.v5n1p62

URL: <http://dx.doi.org/10.5539/cis.v5n1p62>

Abstract

This paper discusses a simulation based file replication for *Malaysian Research and Education Network* (MYREN) Data Grid topology. This simulation evaluates a novel replication strategy, based on an economic model, which optimizes both the selection of replicas for running jobs and the dynamic creation of replicas in MYREN Grid sites. The optimization agents are located on MYREN Grid sites and use an auction protocol for selecting the optimal replica of a data file and a prediction function to make informed decisions about local data replication. Our primary objective of this study is to determine the optimal replication needs to be selecting when given a request by a job for a particular file in MYREN Data Grid implementation. The second objective is to trigger both replication and deletion of data files in MYREN Grid sites by analyzing the pattern of previous file requests; thereby affecting the migration of files toward sites that show a corresponding increased frequency of file-access requests.

In MYREN topology, the major issue is data latency. In order to solve this issue, data replication is considered to be an important technique in reducing jobs execution. Replication involves the creation of identical copies of data files and their distribution over MYREN sites. We evaluate the replication strategy using a Data Grid simulator called OptorSim. Our simulation results showed that the technique can significantly reduce data access latency and increase the robustness of Grid application.

Keywords: Data grid, Replication strategy, Auction protocol, Economic model

1. Introduction

Grid computing is a term as a metaphor for making computer power as easy to access as an electric power grid. Grid computing is an emerging computing model that distributes processing across a parallel infrastructure which reflects a conceptual framework rather than a physical resource. A Grid environment is created to address resource needs. The use of those resources such as CPU cycles, disk storage, data and software programs is usually characterized by its availability outside of the context of the local administrative domain. This external provisioning approach entails creating a new administrative domain referred to as a Virtual Organization (VO) with a distinct and separate set of administrative policies.

Discovery is a part of services in grid computing. It is the process of finding a suitable resource on the Grid to perform required task, such as finding a compute host on which to run a job. This process involves both finding resources that are suitable such as having the correct CPU architecture and then choosing the best resource with the shortest submission queue so that the job will be compute faster. During the computing process, the running jobs mostly will needs to find and access their required data file which is distributed in the system. Here, replication plays a big role in Grid for optimizing the jobs execution.

The rest of the paper is organized as follows: Section 2 provides a brief description of the related works. Section 3 provides the details of the simulation environment and methodology used. Section 4 discusses the design and implementation. Section 5 contains a presentation and analysis of the results. Finally, section 6 lists our conclusions and suggestions for future work.

2. Literature Review

2.1 Data Replication Strategies

Each of the different replication provides customers with choices in business continuance methods. Synchronous replication ensures that both the primary and remote site are synchronized, while asynchronous replication enables customers to choose an acceptable lag time for replicating and restoring data. Asynchronous solutions also provide a greater amount of protection by extending the distance between the primary and secondary locations of the data. Increased distances can provide protection to the local events from the loss of a power grid, and natural disasters such as earthquakes and hurricanes. Each replication process requires similar analysis of the environment. Rates of change, recovery point objectives, and recovery time objectives need to be evaluated in the design of a business continuity solution. Through the use of business Risk Analysis (RA) and Business Impact Analysis (BIA) processes can be established to ensure business is able to continue regardless of an event. In some instances the architect of a disaster recovery solution will require the business process owners to consider alternatives to any given solution enabling the process to meet budget and performance expectations. In (Ungar *et al.*, 2005), the authors outline Sun's approaches to data replication to help customers decide how best to integrate various data replication methods into Business Continuity plans.

2.2 Design and Evaluation of Dynamic Replication Strategies for a High-Performance Data Grid

Physics experiments that generate large amounts of data need to be able to share it with researchers around the world. High performance Grids facilitates the distribution of such data to geographically remote places. Dynamic replication can be used as a technique to reduce bandwidth consumption and access latency in accessing these huge amounts of data. In (Ranganathan & Foster, 2001), authors have described a simulation framework that they have developed to model a grid scenario, which enables comparative studies of alternative dynamic replication strategies. They also presented preliminary results obtained with the simulator, in which they evaluate the performance of six different replication strategies for three different kinds of access patterns. The simulation results show that the best strategy has significantly reduced the latency and bandwidth consumption if the access patterns contain a moderate amount of geographical locality.

2.3 Optimizing Data Replication for Expanding Ring-Based Queries in Wireless Sensor Networks

In (Krishnamachari & Ahn, 2005), the authors consider the problem of optimizing the number of replicas for event information in wireless sensor networks, when queries are disseminated using expanding rings. They obtain closed-form approximations for the expected energy costs of search, as well as replication. Using those expressions, they derive the replication strategies that minimize the expected total energy cost consisting of search and replication costs, both with and without storage constraints. In both cases, they find that events should be replicated with a frequency that is proportional to the square root of their query rates. They have validated their analysis and optimization through a set of realistic simulations that incorporate non-idealities including deployment boundary effects and lossy wireless links.

2.4 Data Replication Strategies in Wide-Area Distributed Systems

Effective data management in today's competitive enterprise environment is an important issue. Data is information; and information is knowledge. Hence, fast and effective access to data is very important. Replication is one such widely accepted phenomenon in distributed environment, where data is stored at more than one site for performance and reliability reasons. Applications and architecture of distributed computing has changed drastically during last decade and so has replication protocols. Different replication protocols may be suitable for different applications. In this manuscript we present a survey of replication algorithms for different distributed storage and content management systems ranging from Distributed Database Management Systems, Service Oriented Data Grids, Peer-to-Peer (P2P) Systems, and Storage Area Networks. In (Geol & Buyya, 2006), the authors discuss in detail the replication algorithms of more recent architectures, Data Grids and P2P systems. They also briefly discuss replication in Storage Area Network and Internet.

From what they have reviewed, Grids computing in a term are viewed as a large scale virtual computer system. There are many issues and part that we have to understand to provide or to establish a Grid environment either in network, physical equipment, middleware or application that run under Grid.

3. Simulation Methodology and Design

This section discusses the technique undertaken to develop the simulation model for MYREN Data Grid environment. Simulation is chosen as a method to investigate the replication algorithm. The simulation studies are performed, not on the real-world system, but on a model of the system (usually computer-based) created for the purpose of studying certain system dynamics and characteristics.

In this study, OptorSim is used to simulate the MYREN Data Grid model. Figure 1 shows the Grid architecture that used in OptorSim (Cameron *et al.*, 2003). The simulation is constructed assuming that the Grid consists of several sites, each of which may provide computation and data-storage resources (called Computing and Storage Elements) for data intensive jobs.

Jobs are submitted to the Grid over a period of time via the Resource Broker (RB). The RB schedules each job to the Computing Elements (CE) with the goal to improve the overall throughput of the Grid. A Replica Manager (RM) at each site manages the data flow between sites and interfaces between the computing and storage resources and the Grid. The Replica Optimizer (RO) inside the RM is responsible for the selection and dynamic creation and deletion of replica file.

Since the Data Grid is a highly dynamic environment, it is important to be able to cope with the changes in the status of resources when performing their allocation to Grid jobs. In particular, we should avoid making irrevocable decisions at job scheduling time about which file replicas will be used to access data for a particular job. Therefore, optimization is performed at three points in time during the life-time of a job:

- The first optimization phase occurs when the RB determines on which CE the job should run. We refer to this phase as Scheduling Optimization.
- In the second phase optimal Dynamic Replica Selection is achieved during the run time of a job via the auction mechanism
- In the final phase, replication can be triggered at third party sites. This is Dynamic Replica Optimization.

3.1 Scheduling Optimization

OptorSim scheduling algorithm takes into account both locality of files requested by jobs and CE loads. The calculation of the file accessing cost is implemented in the RO function `getAccessCost()`. For each CE that is candidate for scheduling, the RB calls `getAccessCost()` Function with a list of files required for the job. The function consults the Replica Catalogue to find all the replicas of each file, and then calculates the time to access each replica from the given CE by examining the current network status. By summing the times to access the “best” replica of each file, `getAccessCost()` Function returns the estimated file access time the job would have if scheduled to the CE.

Since it is concerned with the time to complete a job, i.e. the time the job is submitted to the Grid and the time it has finished executing on a CE, it also take into account the workload of each CE. A combined cost is found by simply adding the file transfer cost obtained from `getAccessCost()` Function to the number of jobs waiting in the queue at the CE.

$$cost = getAccessCost() + estimatedQueuingTime$$

The scheduling algorithm in OptorSim schedules a job to the CE with the minimum combined cost.

3.2 Replica Selection

Once a job is running on a CE, it requires access to files in order to process data. Since there can be multiple copies of each file available on the Grid, the RO function `getBestFile()` is called whenever a file is requested by the job. The function uses the auction mechanism to return the best available replica. Depending on the popularity of the file, replication may also occur if it is deemed to be economically profitable.

3.3 Dynamic Replica Optimization

Grid sites also monitor and log the patterns of file requests over time. If a particular file appears to be popular and a site decides it is likely to make a profit by purchasing it, replication can be triggered to third party sites: those not involved in the initial request for the file. This way the data will migrate towards the areas the data is most likely to be requested.

3.4 An Economy-based Optimization Strategy

OptorSim propose an economic model for data access and replication, to be used in optimization phases 2 and 3. In the model, data files are “purchased” by CEs for running jobs and by Storage Elements (SE) to make an investment that will improve their expected future revenue. These files are sold by SEs to either CEs or other SEs. CEs try to minimize the file purchase cost, while SEs attempt to maximise their profits. CEs and SEs interact with intelligent optimization agents which perform the reasoning required. The adoption of an economic approach has two main motivations. The first reason is to be able to make decision on replica optimization in a distributed manner. Performing such complex multidimensional optimizations in a centralized manner would be difficult, as the domain (attributes of the resources being controlled) is large. By restricting itself to local

optimization through economic interactions it makes the problem manageable and tries to improve the performance through the emergent marketplace behavior. The second reason is that the Data Grid is a highly dynamic environment in which the availability of resources can change without warning. By using an economic model, we can exploit the dynamism of the market to make informed decisions at job execution time.

3.5 Internals of the Economic Model

In the current implementation, file costs are proportional to the time needed to retrieve them, which depends on the available network bandwidth between Grid sites. This way, the goal of minimizing file purchase cost results in an improvement accumulated job execution time. The Grid service that performs optimization of data access and replication is the Replica Optimizer (RO). It is a distributed service and is provided by a set of RO agents, one for each Grid site (see Figure 2). A RO agent is invoked by a Grid job, running on a local CE that needs to access a data file. The RO is able to select the cheapest replica of that file, possibly triggering file replication between Grid sites if this is predicted to improve Grid performance. The abstract architecture for RO agents includes the following components:

- *Access Mediator (AM)*. This component processes file requests from jobs running on a CE. For each requested file, it starts an auction to identify the cheapest replica of the file. The AM gathers bids for the file from local and remote Storage Brokers (SB), selects the winner of the auction and notify the bidders about the winner.
- *P2P Mediator (P2PM)*. These are responsible for establishing and maintaining a peer to peer communications infrastructure between Grid sites. They propagate auction messages between AMs and SBs.
- *Storage Broker (SB)*. This component is responsible for listening for file request messages from the local P2P Mediator. If it can meet the request with a file stored in the corresponding SE, it responds immediately to the P2P by starting a nested auction in order to obtain a local replica of the file and be able to reply to the parent auction. Since a Grid site might exclude CEs or SEs, some of the above components might be absent from the available RO agents for that site. This is depicted in Figure 2. The figure shows an example Data Grid containing four sites. Site 3 contains both Computing and Storage Elements and thus the local RO agents comprise of all components. Other sites include only the components needed and thus the local RO agents are simpler

4. Design and Implementation

This section discusses in detail about design and implementation phase of the project. In this project, we simulate the MYREN topology which is launched by Ministry of Energy, Water and Communication (MEWC), Ministry of Science and Technology (MOSTI) and Ministry of High Education (MOHE) to enable high-speed dedicated network connectivity for research and educational application. The OptorSim simulator is used to evaluate an economy-based file replication strategy for MYREN Data Grid model. The simulator was initially developed by the EU DataGrid, with major contributions from GridPP (Bell *et al.*, 2003). It can be used for testing file replication and job scheduling algorithms. By using OptorSim simulator, we can easily set up a Grid topology and list of jobs to run, choose a job scheduler and replica optimization strategy and study the performance of the chosen algorithms.

4.1 Test-bed Topology

The MYREN topology is shown in Figure 3. In MYREN topology, it is consist of 12 largest public universities in Malaysia as the sites which virtually connected to each other as star topology with a given bandwidth ranging from 2 Mbps to 8 Mbps. As shown in Figure 3, MIMOS router are the center of all sites before connected to main site, MYREN NOC. MYREN NOC is the main for all sites which all the jobs were executed. MYREN's router located in Cyberjaya and is a gateway to other international NRENs in other countries via the Trans-Eurasia Information Network (TEIN2). Countries currently connected to MYREN include Asia Pacific - Australia, China, Indonesia, Japan, Korea, Singapore, Philippines, Thailand, and Vietnam. Europe – GEANT2 (33 countries in Europe include UK, Germany, Italy, Spain, etc.). North America – Internet2.

4.2 Resources Information

All the sites information, bandwidth, and connection are the important data used and implemented to the simulation as shown in Table 1. The existing topology used in OptorSim is simulated as the MYREN topology and all the information gathered are used in the simulation. All the information has to be accurate for better result at the end.

4.3 Configuration Files

After all the data and information gathered, the next important is creating the configuration files which is suitable with the topology chosen. This configuration file will be used when running the simulator. There are several configuration files used to control various inputs to OptorSim. The Grid Configuration File describes the Grid topology and the content of each site; that is, the resources available and the network connections to other sites.

The Job Configuration File contains information on the simulated files, jobs and the site policies for each site (the list of files each site will accept). The Simulation Parameters File contains various simulation parameters which the user can modify. If the user wishes to simulate background network traffic, a Bandwidth Configuration File is needed; along with several data files to describe the simulated traffic. A further configuration file, the GUI Configuration File, is used if the simulation is to be visualized by the inbuilt OptorSim GUI. Sample configuration files for OptorSim can be found in the examples/directory. When using the supplied examples are used, the grid and job configuration files must match up, for example mygrid_grid.conf and mygrid_jobs.conf.

4.3.1 The Grid Configuration Files

This file describes the status of the resources of each site and the layout of the simulated Grid. The example configuration file shown in Figure 4 describes the test-bed grid used which is the MYREN Grid. Each row in the configuration file is the information for one site:

As shown in Figure 4, the first row in the grid configuration file corresponds to site 1 in MYREN Grid topology. All this sites has a CE and a SE with a size of 50000 MB. It is linked to all sites in MYREN Grid topology and each bandwidth ranging from 2 to 45 Mbps.

- The first column shows the number of worker nodes in the Computing Element (CE) at the site (if one is present).
- The second column is the number of Storage Elements (SEs) at the site.
- The third column is the size in MB of the SE, if one is present.
- The rest of the table is a site vs. site matrix giving the maximum bandwidth (i.e. link capacity) between each site in Mbps. The matrix is diagonally symmetric. Entries along the diagonal are ignored since network bandwidth within a site is assumed to be infinite.

4.3.2 Job Configuration File

This file contains information about the files present on the Grid and the jobs submitted during the simulation. It is shown in Figure 5. It is very important that the CEs that are assigned jobs match up with the CEs declared in the Grid configuration file.

- The first part (included between *begin{filetable}* and the first end) is a list of file names, their sizes in MB and their unique numerical identifiers. Each file name is a Logical File Name and is unique. The file IDs are used by a particular optimization algorithm.
- Each row in the second part of the job configuration file (included between *begin{jobtable}* and the second end) contains a job name and the list of files required for the job.
- Each row in the third part of the job configuration file (included between *begin{cescheduletable}* and the third end) gives the scheduling policy for each CE i.e., which jobs each CE is willing to run.
- The fourth part of the job configuration file, which is included between *begin{jobselectionprobability}* and the fourth end, determines the frequency with which each job are submitted to the Grid. In Figure 5, job1 has a 50% probability of being submitted whereas job2 has a 30% probability.

5. Results and Discussions

In this section we present simulation results, using OptorSim to evaluate and compare the optimization strategies for both scheduling and run-time optimization. Some of the measurements are used as indicators of how well the strategy performs. We start the evaluation by studying the impact of the scheduling algorithm used by the Resource Broker (RB) on a given optimization strategy. The following scheduling algorithms are analyzed:

- *Random*: scheduling is done randomly to any Computing Element that will run the job.
- *Queue Length*: schedules to the Computing Element with the shortest queue of waiting jobs. If two CEs have the same shortest queue length one of them is chosen at random.

- *File access cost*: schedules to the Computing Element from which the cost to access all the files required for the job (in terms of network latencies) is the smallest. If two CEs have the same smallest access cost one of them is chosen at random.
- *File access cost + job queue access cost*: scheduling is done using a combination of the access cost for the files and the access cost for all the jobs in the queue at each Computing Element.

The simulation runs 1000 jobs and for each scheduling algorithm considering each of the three access patterns (sequential, Gaussian random walk and Zipf). The results of the mean job time and CE usage for the three optimisation strategies and the three access patterns are shown in Figure 6a, 6b, 7a, 7b, 8a and 8b. The Random and Queue Length algorithms show similar performance and generally have the longest mean job times. The access cost for current job algorithm has a lower mean job time but has the lowest CE usage, due to the fact that jobs are only scheduled to sites with high network connectivity.

The mean job time decrease and CE usage is increasing when we use the access cost for current job + all queued jobs algorithm. This gives the best balance between scheduling jobs close to the data whilst ensuring sites with high network connectivity are not overloaded and sites with poor connectivity are not idle. The SE usage for each of the above scheduling strategies was also monitored as the simulation progress, considering the three optimization strategies with sequential access patterns. Figure 9a shows that the Random and Shortest Queue strategies quickly filled up all the available SEs to reach the maximum of 90% probability, while Access Cost for current job only used the SEs with high network connectivity, resulting in slower execution time and a final SE usage level of only 37% probability. Access Cost for current job + all queued jobs, on the other hand, took network connections into account but also avoided long queues of jobs, resulting optimal SE usage and fast execution time.

All optimization strategies gave very similar results, as can be seen for two of the schedulers in Figure 9b. The graph shows the differences between Access Cost for current job algorithm and Access Cost for current job + all queued jobs algorithm. Note that 100% SE usage is never reached to the Grid configuration used with Access Cost for current job + all queued jobs. In conclusion, the economic model shows significant performance improvements for sequential access patterns. Even though for Gaussian walk access patterns LRU performs better, this simulation shows that the economic model is more robust against jobs accessing files with various patterns.

6. Conclusion

The paper presented a novel optimization technique, based on an economic model, and discussed its use in a Data Grid to optimize both replica selection and dynamic access-pattern-driven replication. It's detailed the reverse Vickrey auction protocol that the optimization agents use for dynamically selecting the best replica of a requested file. This considers both data locality and network latencies. It is also discussed the prediction function the agents use for making replication decisions, which uses historical data about file access patterns.

To evaluate the efficiency of our algorithm we ran the Grid simulator OptorSim configured to represent a real world Data Grid test-bed from MYREN, using job throughput as a benchmark. This paper also studies on the impact of different file access patterns on the performance of the economic model and compared our results with traditional replication algorithm. The results clearly show that, for some access patterns, the economic model based replication algorithm shows a markedly improved performance compared to the traditional algorithm. This is due to its ability to prioritize files based on file access history and to replicate accordingly.

As part of our future work, we plan to extend this simulation framework and evaluate the results against typical access patterns from other large-scale analysis efforts. Moreover, we also plan to embed in the economic model the file payment and take into consideration also storage costs, which have been neglected so far.

References

- Asadzadeh, P., Buyya, R., Kei, C., L., Nayar, D., & Venugopal, S. (2005). Global Grids and Software Toolkits: A Study of Four Grid Middleware Technologies. *High Performance Computing: Paradigm and Infrastructure*, Laurence Yang & Minyi Guo (eds), 431-458 (Chapter 22). Wiley Press, New Jersey, USA.
- Bell, W., H., Cameron, D., G., Carvajal-Schiaffino, R., Millar, A., P., Stockinger, K., & Zini, F. (2003). *Evaluation of an Economy-Based File Replication Strategy for a Data Grid*. CERN, European Organization for Nuclear Research, Switzerland.
- Buyya, R., & Murshed, M. (2002). *GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing*. Monash University, Australia.

- Cameron, D., G., Carvajal-Schiaffino, R., Millar, A., P., Nicholson, C., Stockinger, K., & Zini, F. (2003). *OptorSim: A Grid Simulator for Replica Optimisation*. Lawrence Berkeley National Laboratory, USA.
- Foster, I. (2002). *File and Object Replication in Data Grids*. ACM.
- Foster, I. (2002). *What is the Grid? A Three Point Checklist*. Argonne National Laboratory & University of Chicago, US.
- Foster, I., & Kesselman, C. (1998). *The Grid: Blueprint for a New Computing Infrastructure*. Univ. of Chicago, Chicago.
- Goel, S., & Buyya, R. (2006). *Data Replication Strategies in Wide Area Distributed Systems*, Enterprise Service Computing: From Concept to Deployment. Robin G. Qiu (ed), 211-241. Idea Group Inc., Hershey, PA, USA,
- Krishnamachari, B., & Ahn, J. (2005). *Optimizing Data Replication for Expanding Ring-based Queries in Wireless Sensor Network*. Department of Electrical Engineering, University of Southern California.
- Lin, H., Abawajy, J., & Buyya, R. (2006). *Economy-Based Data Replication Broker*. Proceedings of the 2nd IEEE International Conference on E-Science and Grid Computing (E-Science 2006). Amsterdam.
- Ranganathan, K., & Foster, I. (2001). *Design and Evaluation of Dynamic Replication Strategies for a High-Performance Data Grid*. Department of Computer Science, University of Chicago.
- Ranganathan, K., & Foster, I. (2002). Decoupling Computation and Data Scheduling in Distributed Data-Intensive Applications. *Math and Computer Science Division*, Argonne National Laboratory, USA.
- Ungar, R., Thomas, T., Brouwer, P., Wilson, S., & Threlfall, M. (2005) *Data Replication Strategies*. Sun Microsystems, California, USA.

Table 1. Bandwidth capacity for each site

Site ID	Site Name	Bandwidth(Mbps)	Site ID	Site Name	Bandwidth(Mbps)
1	MYREN NOC	45	8	MMU	2
2	USM	8	9	UNITEN	2
3	UUM	2	10	UM	8
4	UTP	4	11	UTM	8
5	UPM	8	12	UMS	1
6	UKM	8	13	UNIMAS	8
7	UiTM	2			

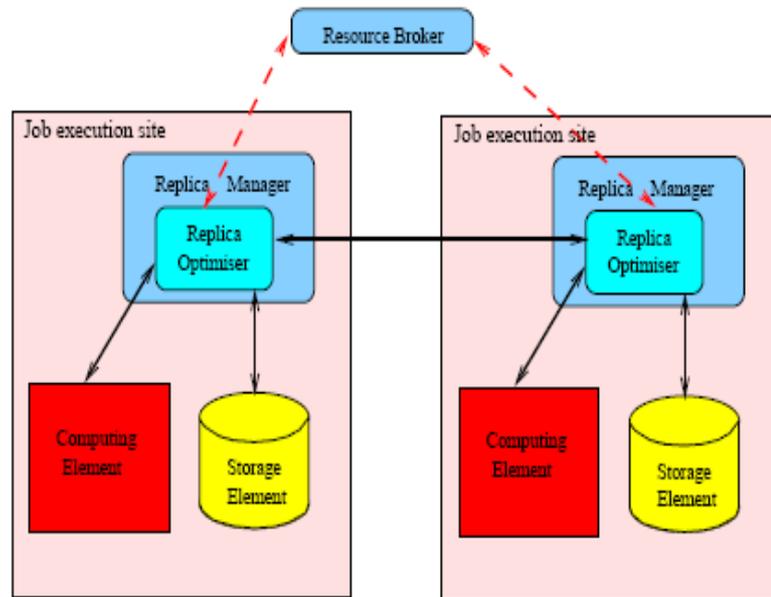


Figure 1. Simulated data grid architecture (Cameron *et al.*, 2003)

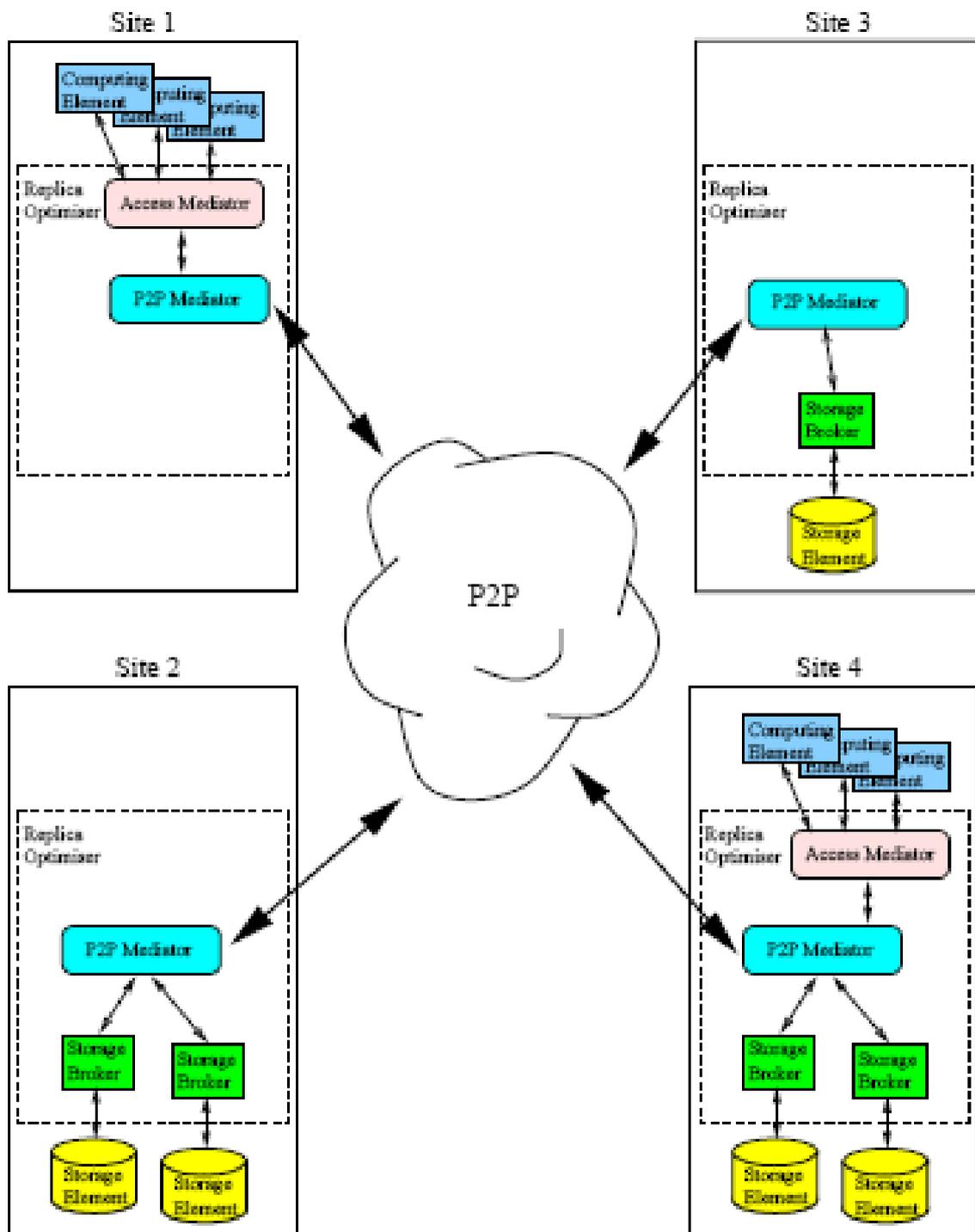


Figure 2. Components of the economy-based grid model (Cameron *et al.*, 2003)

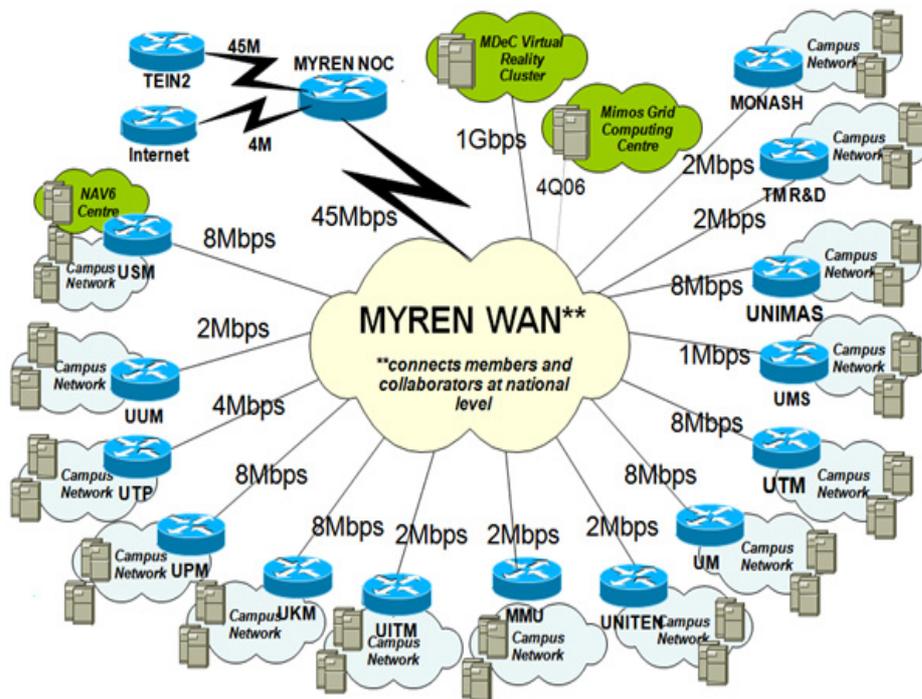


Figure 3. MYREN network topology

```
0 1 50000 0. 45. 8. 2. 4. 8. 8. 2. 2. 2. 8. 8. 1. 8.
1 1 10000 45. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0 1 10000 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

Figure 4. Grid configuration file

```
\begin{filetable}
File1 1000 1
File2 1000 2
File3 1000 3
File4 1000 4
File5 1000 5
File6 1000 6
File7 1000 7
File8 1000 8
File9 1000 9
File10 1000 10
\end
#
# Job Table
# A job name and a list of files needed.
#
\begin{jobtable}
job1 File1 File2 File3 File4 File5 File6
job2 File7 File8
\end
#
# CE Schedule Table
# CE site id, jobs it will run
#
\begin{cescheduletable}
1 job1
\end
#
# The probability each job runs
#
\begin{jobselectionprobability}
job1 0.5
job2 0.3
\end{jobselectionprobability}
```

Figure 5. Job configuration file

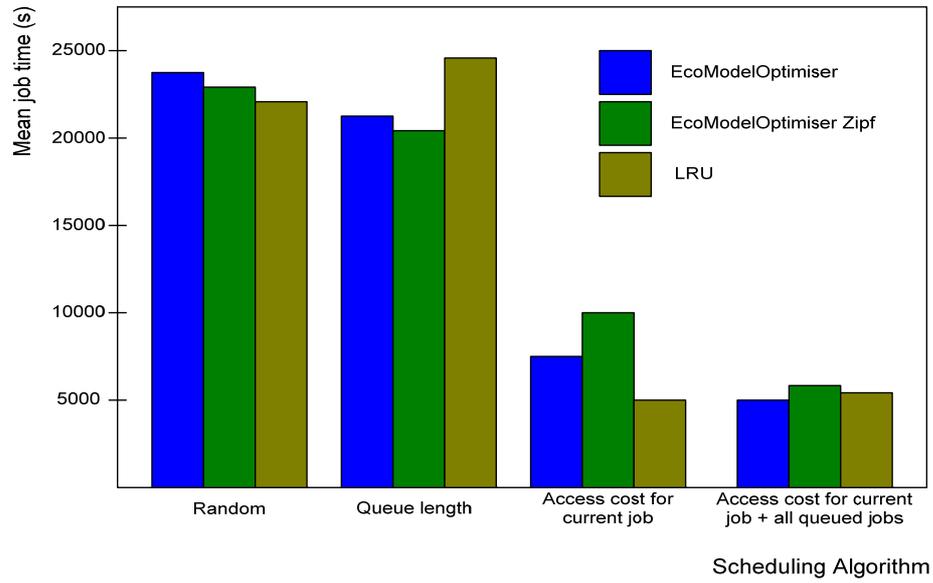


Figure 6a. Mean job time for various optimisation algorithms and sequential access pattern (SequentialAccessGenerator)

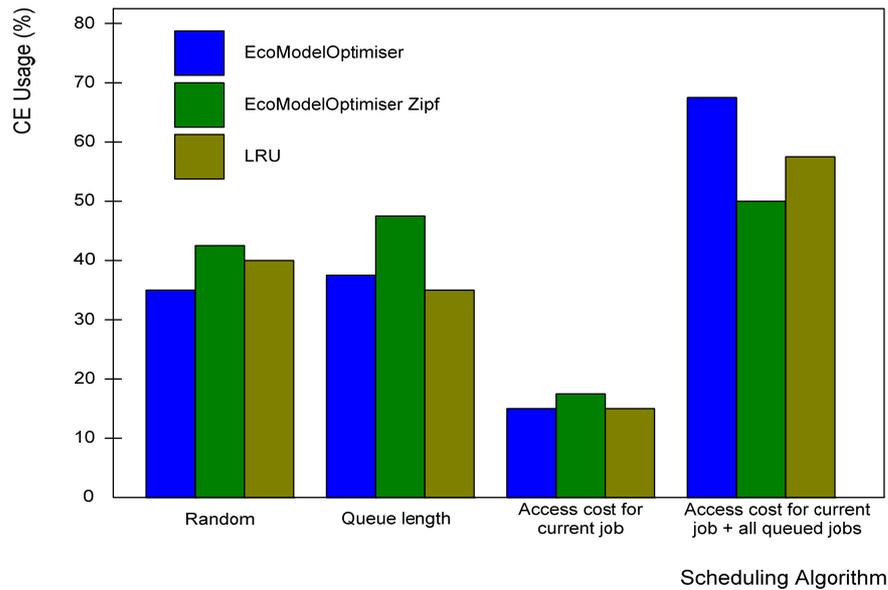


Figure 6b. CE usage for various optimisation algorithms and sequential access pattern (SequentialAccessGenerator)

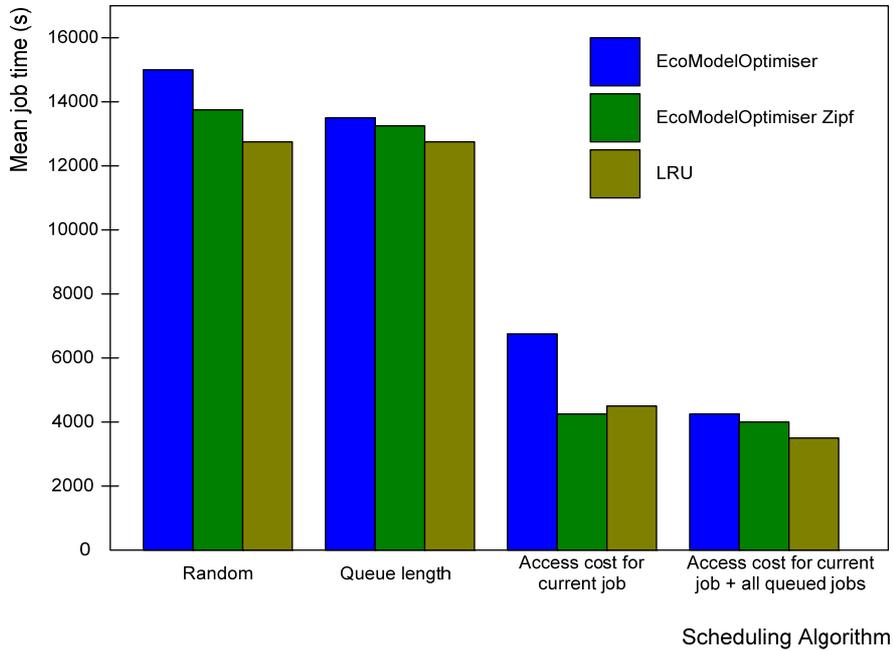


Figure 7a. Mean job time for various optimization algorithms and Gaussian random walk access pattern (RandomWalkGaussianAccessGenerator)



Figure 7b. CE usage for various optimization algorithms and Gaussian random walk access pattern (RandomWalkGaussianAccessGenerator)

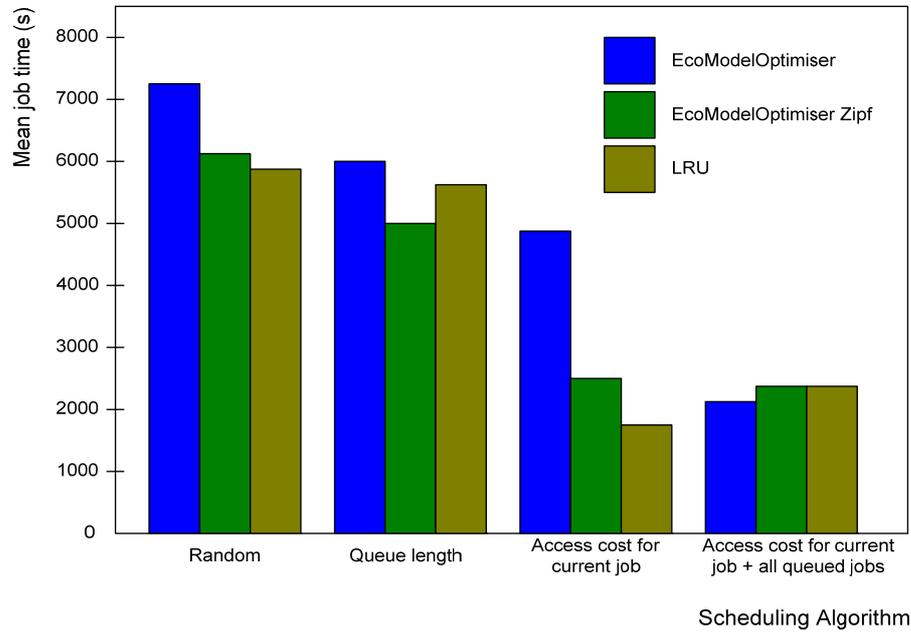


Figure 8a. Mean job time for various optimization algorithms and Zipf access pattern (RandomZipfAccessGenerator)

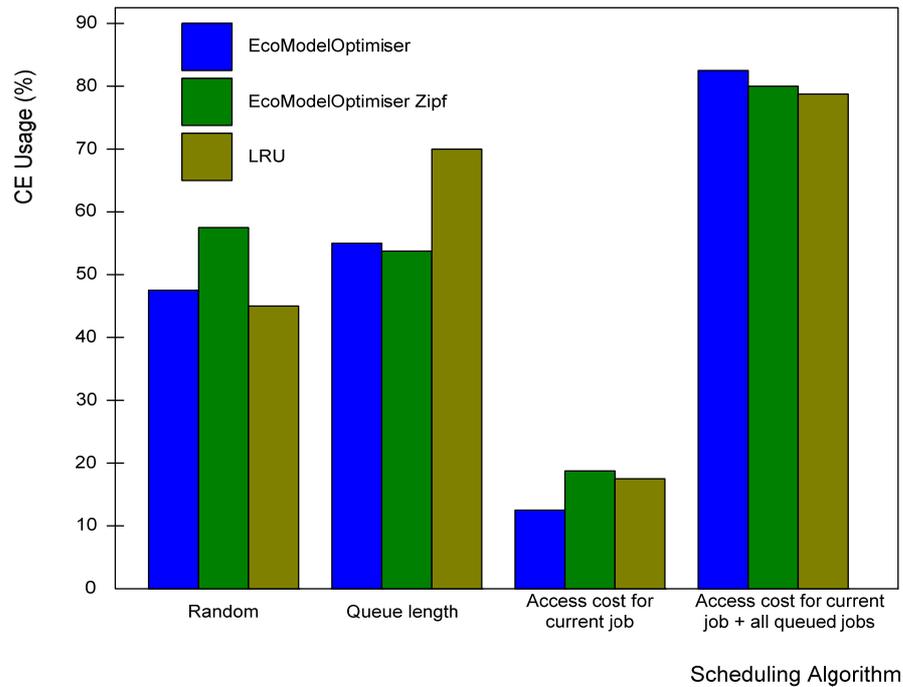


Figure 8b. CE usage for various optimization algorithms and Zipf access pattern (RandomZipfAccessGenerator)

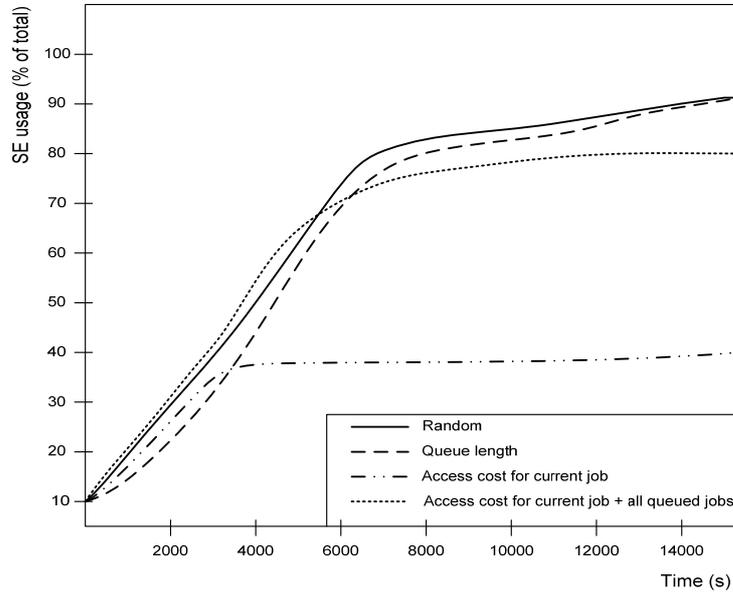


Figure 9a. SE usage for all scheduling algorithms

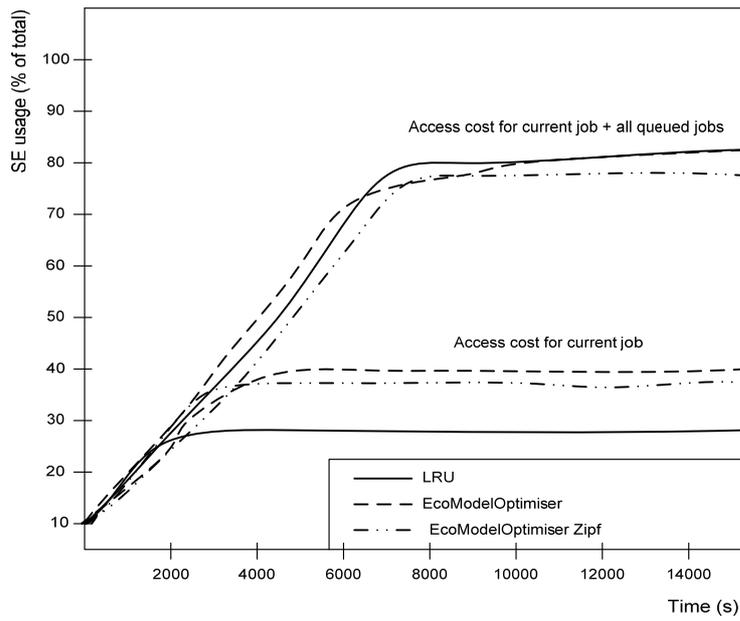


Figure 9b. SE usage for all optimization strategies, Access Cost for current job + all queued jobs and Access Cost for current job schedulers