Evaluating Scheduling and Replica Optimisation Strategies in OptorSim

David G. Cameron[1], Rubén Carvajal-Schiaffino[2],
A. Paul Millar[1], Caitriana Nicholson[1], Kurt Stockinger[3], Floriano Zini[2]

[1] University of Glasgow, Glasgow, G12 8QQ, Scotland

[2] ITC-irst, Via Sommarive 18, 38050 Povo (Trento), Italy

[3] CERN, European Organization for Nuclear Research, 1211 Geneva, Switzerland

Abstract

Grid computing is fast emerging as the solution to the problems posed by the massive computational and data handling requirements of many current international scientific projects. Simulation of the Grid environment is important to evaluate the impact of potential data handling strategies before being deployed on the Grid. In this paper, we look at the effects of various job scheduling and data replication strategies and compare them in a variety of Grid scenarios, evaluating several performance metrics. We use the Grid simulator OptorSim, and base our simulations on a world-wide Grid testbed for data intensive high energy physics experiments.

Our results show that the choice of scheduling and data replication strategies can have a large effect on both job throughput and the overall consumption of Grid resources.

1 Introduction

In a Data Grid, replication of data is critical for maximising overall job throughput. Replication involves creating copies of data files at different sites on the Data Grid. *Replica Optimisation* strategies define when replicas should be created or deleted on a per-site basis, and which replicas should be used by Grid jobs. These decisions take into account a job's usage of three resources: computational power, data storage and network bandwidth.

The EU DataGrid project [18] is building computing infrastructure and middleware services for the management of large-scale data across widely distributed scientific communities. Within this project a *Replica Optimisation Service* (ROS) [3] is being developed to address the issues related to Replica Optimisation.

Before deploying the ROS it is important to select one or more optimisation strategies it will use. These strategies must work effectively in a Data Grid under a wide range of conditions, so should be thoroughly tested before they are employed. One way to achieve a realistic evaluation of various strategies is to define a Grid simulation environment that closely mimics a real Data Grid. This environment should be capable of simulating a number of Grid jobs using a candidate optimisation strategy, and to collect measurements on which the evaluation of the strategy is based. The authors have been developing the Grid simulator OptorSim [4, 17], which has been used to evaluate several Grid replication strategies. In particular, an auction protocol and economic model for replica optimisation were introduced [5] and it was shown that for certain file access patterns this model significantly outperforms more traditional models.

In this paper, we first discuss the key elements of a realistic Grid model, which forms the basis of our simulation environment. With respect to our previous work, we consider the effects of different job scheduling strategies and improve the accuracy of the model by taking into account background (i.e. non-Grid) network traffic, which can use a sizable proportion of the underlying network resources. Second, we analyse some important metrics which are useful for the evaluation of a Data Grid. In previous experiments [5], the metric used was the total execution time of a set of jobs on the Grid. Here, we add other indicators of performance that are significant for different types of Grid user. Third, we present the performance of a new optimisation strategy which uses a Zipf-based prediction function [22] to evaluate the relative worth of files using previous file access patterns.

The paper is structured as follows: related work on Grid simulations is examined in Section 2; we then discuss the features of a realistic simulation environment in Section 3. Section 4 describes our scheduling and replica optimisation strategies in detail, and a set of performance measurements to evaluate these strategies is presented in Section 5. The specific setup we use for our simulations is described in Section 6 and results are presented in Section 7. Finally, we summarise and draw some conclusions in Section 8.

2 Related Work

Various Grid simulation projects have been undertaken in recent years, among them ChicagoSim [15] [16], EDGSim [9], GridSim [7], and GridNet [11].

In [15] the authors simulated various replication and caching strategies in a tier-model Grid environment. In [16] they combined replication strategies with different scheduling algorithms and found that when using any replication strategy, taking into account the location of data when scheduling vastly improves the overall job performance. They found, however, that there was no marked difference in the choice of replication algorithm, perhaps because replication took place at the level of entire file sets (one file set defining a job) rather than individual files.

EDGSim [9] was designed to simulate the performance of the EU DataGrid but concentrates on the optimisation of scheduling algorithms. Analysis showed that data location was important in the scheduling decision, but no replication of data was taken into account.

The GridSim project [7] is very detailed in its simulation of the components of a Grid and introduces an economic model to manage the use of Grid resources through the buying and selling of resources. It is designed primarily to study scheduling algorithms and does not examine the issue of data replication.

In [11] a replication algorithm is tested which uses a cost function to predict whether replicas are worth creating. It is found to be more effective in reducing average job time than the case where there is no replication. The simulation architecture used was based on a hierarchical model where leaf client nodes ran jobs but higher nodes contained all the storage resources, in contrast to the EU DataGrid architecture which we describe in Section 3.

The main advantage of OptorSim [4, 17] with respect to the previous simulators is that it performs two-stage optimisation. Scheduling decisions are based on both the location of data and the status of network links between grid sites, while (re)optimisation during the run-time of a job takes into account dynamic variations in the distribution of data and in the behaviour of network resources.

3 A Realistic Simulation Environment

A realistic Grid simulation environment should be based on a Grid model that represents a real Data Grid at the proper level of abstraction. This is true for our model, which also includes the elements for describing the Grid topology and site structure, the set of jobs to be simulated, and the parameters that regulate the dynamics of the simulation. In the following we focus on the various components of our model.

3.1 Components of the Simulation Model

Grid Architecture. We adopt a Grid structure based on a simplification of the architecture proposed by the EU DataGrid project [18], as illustrated in Figure 1. In this

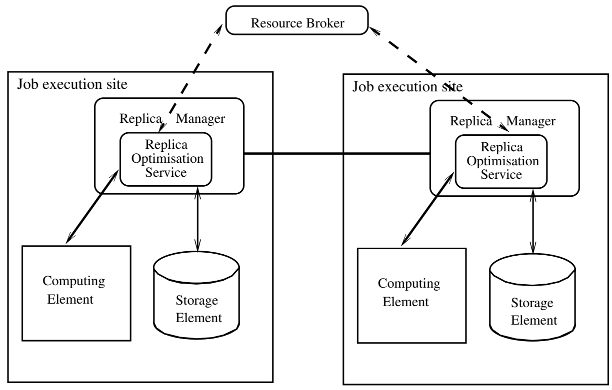

Figure 1. European DataGrid Architecture.

model, the Grid consists of several *sites*, each of which may provide computational and data-storage resources for submitted jobs. Each site consists of zero or more *Computing Elements* (CEs) and zero or more *Storage Elements* (SEs). Computing Elements run jobs that use the data in files stored on Storage Elements. A *Resource Broker* controls the scheduling of jobs to Computing Elements, with the aim of improving the overall throughput of the Grid. In the simulation, sites without Storage or Computing Elements act as network nodes or routers. Grid sites are connected by *Network Links*, each of which has a certain bandwidth. A *Replica Manager* at each site manages the data flow between sites and interfaces between the computing and storage resources and the Grid. The *Replica Optimisation Service* [3] inside the Replica Manager is responsible for both replica selection and the automatic creation and deletion of replicas.

Files and Jobs. In our model a data file is characterised by its name, size and an index number which represents its similarity to other files. A job is specified by the set of data files it needs to analyse. As we are interested in the effect of different optimisation strategies, files are considered homogeneous and only the movement of files caused by replication is simulated, not the analysis processes performed by the jobs. We also model the initial distribution of files to Data Grid sites before starting a simulation run.

Dynamics. The order in which a job requests files is determined by the *Access Pattern* used. Several different access patterns have been chosen for the simulation, allowing the simulation of several types of Grid job. Another important aspect is background network traffic, which can vary unpredictably over time. Any replication strategy must be sufficiently flexible that it can adapt to utilise the continually changing environment, obtaining the best performance for its users.

3.2 OptorSim

To evaluate optimisation strategies in a realistic simulation scenario we have developed the Grid simulator OptorSim [17]. Given (a) a Grid topology and resources, (b) a set of jobs that the Grid must execute and (c) an optimisation strategy, OptorSim simulates what would happen in the Grid if that optimisation strategy were in use. It provides us with the set of measurements described in Section 5 in order to quantify the effectiveness of the strategy.

The design of OptorSim follows the model described in Section 3.1. In addition to the components shown in Figure 1, it also models a peer-to-peer structure within the Replica Optimiser which is used by the auction protocol for file selection we presented in [5].

4 Optimisation Strategies

In this paper we will consider two stages of optimisation, namely *scheduling optimisation* (deciding where a job should be executed) and *run-time optimisation* (deciding which is the best replica for a file request and how best to position the data). We do not consider intra-site optimisation, for example in the scheduling of jobs to worker nodes within a Computing Element or internal queueing systems for jobs – the focus is on the overall optimisation of Grid resources rather than performing optimisation within each element.

4.1 Scheduling Optimisation Strategies

Much work has been done on scheduling optimisation in a Grid and indeed, most Grid simulators make it their central theme (see Section 2). In this paper we will compare several scheduling algorithms, some of which use the location of data to find the best site on which to run a job.

The Resource Broker uses a scheduling algorithm to calculate the cost of running a job on a group of candidate sites. It then submits the job to the site with the minimum estimated cost. The algorithms we test are based on one or more of the following cost metrics:

- *Access Cost*. The estimated cost, based on the current network status, for obtaining all the files required by the job. This metric uses a *Replica Catalogue* (a Grid service, implemented in OptorSim as a table containing the locations of every copy of every file) to look up all the replicas for each required file. The access time for each replica is calculated, thus the best replica can be found for each file. The combined access time for the best replicas is used to rank candidate sites.
- *Queue Size*. The number of jobs waiting in the queue at the candidate site. We assume only one job at a time can run on each CE.
- *Queue Access Cost*. For each job in the queue the access cost is calculated as for the *Access Cost* algorithm. The access costs for all jobs are summed to give a total estimated access cost for all the jobs in the queue.

4.2 Run-time Optimisation Strategies

We assume that run-time optimisation is performed in a distributed way by a number of *Replica Optimisation Agents* (or *Optimisers*), one for each Grid site. An Optimiser performs local replica optimisation; the aim is to achieve global optimisation as the emergent result of local optimisation, so every Optimiser has two goals:

- *To minimise a single job's execution cost.* Users want their jobs to be executed with minimum cost; therefore, an Optimiser aims to minimise the execution cost of every job that is executed in its Grid site. In this paper we define cost to be the total running time of the job.
- *To maximise the usefulness of locally stored files.* Good utilisation of available resources is another goal of optimisation. An Optimiser should therefore also aim to keep locally those files that are most useful for jobs that are executed either locally or at neighbouring sites with good network connectivity.

Whenever an Optimiser is considering a file request, it executes the following tasks:

- *Replication Decision*. If a requested file is not present on a site's SE, this process decides whether local replication of this file should take place. If the Optimiser decides not to replicate a file then the job must remotely access that file.
- *Replica Selection*. Whenever considering remote replicas, this process selects the best of those available. In general, the selection criterion depends on the chosen evaluation measure (see Section 4.3).
- *File Replacement*. When a remote replica has been selected for replication to the site's SE, the SE might not have sufficient spare capacity. In this case, one or more

local replicas must be deleted. The selection criteria for deciding which locally stored replicas to delete depend on some estimate of future usefulness.

A specific combination of algorithms for each stage defines a run-time replica optimisation strategy.

4.3 Specific Run-time Optimisation Strategies

We consider three specific optimisation strategies: one based on the traditional LRU (Least Recently Used) algorithm, and two economic strategies.

The LRU-based strategy will always replicate files to storage local to the Computing Element on which the job is running. Replica Selection is achieved by using looking up a Replica Catalogue to locate all replicas. The replica that can be accessed in the shortest time, under the current network conditions, is chosen. If the local storage is full, the file that has been accessed the fewest times in the previous time window is deleted, creating space for the new replica.

The two economy-based strategies are similar to each other, but use two different prediction functions, one binomial-based [8] and the other Zipf-based, to calculate file values used in the replication and file replacement decisions. If the potential replica under consideration has a higher value than the lowest-valued file currently in the local storage, that file is deleted and the new replica is "bought". If local storage is not yet full, the economic models will always replicate.

The file value is approximated by the number of times it is expected to be accessed in a future time window $\delta T'$, based on the file access history for the previous time window δT. The binomial prediction function constructs a binomial distribution of file popularity, centred on the mean number of file accesses in δT. The value of the file in question is then found by checking where it lies on that distribution. This prediction function is described in more detail in [8]. The Zipf prediction function orders the files into a Zipf distribution according to their popularity in δT, and takes the value from there. A description of the Zipf distribution function is given in Section 6.

Replica Selection is based on the auction protocol for buying and selling files described in [5].

5 Evaluation of Grid Optimisation Strategies

There are several measures which can be considered in the evaluation of Grid optimisation strategies. Since the effectiveness of a strategy could depend on the adopted measure, a discussion of a number of measures can assist in the choice of a suitable strategy. We have identified the following measures, provided by OptorSim, as being useful for strategy evaluation:

Mean Job Execution Time. The mean job execution time is defined as the total time to execute all the jobs, divided by the number of jobs completed. This is related to the measure we used in [4, 5] and a typical Grid user would probably consider it to be the most important measure of how the algorithm is performing.

Network Usage. Replicating a file takes time and uses network bandwidth. However, performing no replication has been shown [4] to be ineffective compared to even the simplest replication algorithm. Thus, a good balance must be found, where any replication is in the interest of reducing future network traffic. We define *effective network usage* r_{ENU}:

$$r_{\mathrm{ENU}} = \frac{N_{\text{remote file accesses}} + N_{\text{file replications}}}{N_{\text{local file accesses}}},$$

where $N_{\text{remote file accesses}}$ is the number of times the CE reads a file from a SE on a different site, $N_{\text{file replications}}$ is the total number of file replications, and $N_{\text{local file accesses}}$ is the number of times a CE reads a file from a SE on the same site (we assume infinite bandwidth within a site).

For a given network topology, a lower value of r_{ENU} indicates the optimisation strategy is better at replicating files to the correct location.

SE Usage. Monitoring the use of storage resources in Grid sites can also be a valuable source of information and to this end, we can measure the percentage of storage used during the simulation. This can help in evaluating a strategy from two opposing points of view: on the one hand, the goal could be the minimisation of storage usage, perhaps because the resource cost is proportional to the amount being used; on the other hand, its cost might be fixed and one would then aim at maximising the use of storage space.

CE Usage. Another resource which is of interest is the computational power usage, which we define as the percentage of time that a CE is running jobs or otherwise active. Henceforth, we use the term CE usage, which is the total computational power usage for all the CEs on the Grid. A good scheduler should be able to maximise the CE usage by spreading the workload, avoiding the situation where some sites lie idle while others have long queues of jobs.

6 Simulation Setup

In this section we describe how we set up the simulation to model a realistic Grid environment.

Grid Topology. In this paper we base all simulation studies on the testbed used during a large scale production effort for the high energy physics experiment CMS [12]. We used a similar testbed configuration in our previous work [5]. The Grid topology (see Figure 2) comprises 20 sites in Europe and the USA. CERN and FNAL (where the data are originally produced) have a storage capacity of 100 GB each and a master copy of each file is stored at one of these two sites. Every other site has a Computing Element and initially empty storage of capacity 50 GB. The storage capacity values used are representative, having been scaled down from the actual resources at these sites. During our simulation studies we include contending network traffic which is based on network monitoring data between some of the sites shown in Figure 2 (see Section 7.3).

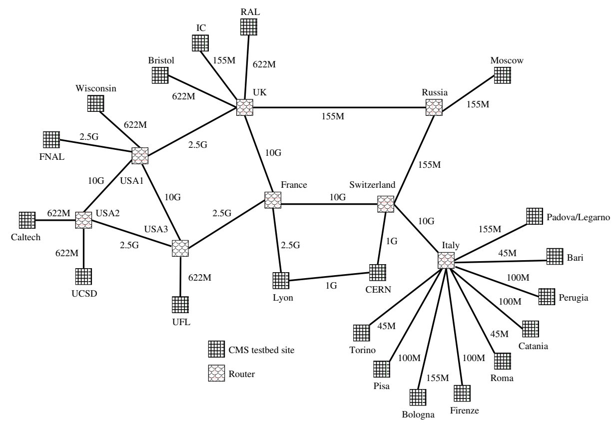

Figure 2. Grid topology for CMS world wide data production challenge in spring 2002.

Simulated Jobs. The simulated work loads are based on a scaled down set of high energy physics analysis jobs from the CDF experiment use case (as described in [10]). In this case each file has a size of 1 GB and the total size of the file set is 97 GB. Each CE takes one second to process each file.

Access Patterns. In this paper we will consider the following access patterns: *sequential* [4] (all files are requested in a predetermined order), *Gaussian random walk* [4] (successive files are selected from a Gaussian distribution centred on the previous file) and *Zipf*.

A Zipf-like distribution can be regarded as a special kind of exponential distribution which is defined as $P_i \propto i^{-\alpha}$, where P_i is the frequency of occurrence of the i^{th} ranked item and $\alpha \lesssim 1$. In other words, a few items in the observed set occur very often while many others occur rarely.

Due to the increasing importance of the web as an Internet application, it has become necessary to model and reproduce typical web workloads [1, 2] in an accurate way. Web proxy caching techniques especially have received considerable interest due to the importance of reducing web traffic [6]. Much of this research shows that the distribution of requested pages generally follows a Zipf-like distribution and it was therefore decided to investigate the effects of such a distribution in a Grid environment.

7 Results

In this section we present simulation results, using OptorSim to evaluate and compare the optimisation strategies for both scheduling and run-time optimisation. Some of the measurements described in Section 5 are used as indicators of how well the strategy performs.

7.1 Access Patterns and Scheduling

We start our evaluation by studying the impact of the scheduling algorithm used by the Resource Broker on a given optimisation strategy. The following scheduling algorithms are analysed (see Section 4.1):

- *Random:* Schedule to a random CE.
- *Shortest Queue:* Schedule to the CE with the shortest job queue.
- *Access Cost:* Schedule to the CE where the job has lowest file access cost.
- *Queue Access Cost:* Schedule to the CE where the sum of the access cost for the job itself and the access costs of all jobs in the queue is smallest.

We ran the simulation with 1000 jobs submitted at 5 second intervals, and for each scheduling algorithm we considered each of the three access patterns described previously (*sequential*, *Gaussian random walk* and *Zipf*).

Results showing the mean job time and CE usage for the three optimisation strategies and the three access patterns are shown in Figures 3, 4, and 5.

The *Random* and *Shortest Queue* algorithms show similar performance and generally have the longest mean job times. The *Access Cost* 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 is lowest and CE usage is highest when we use the *Queue Access Cost* 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.

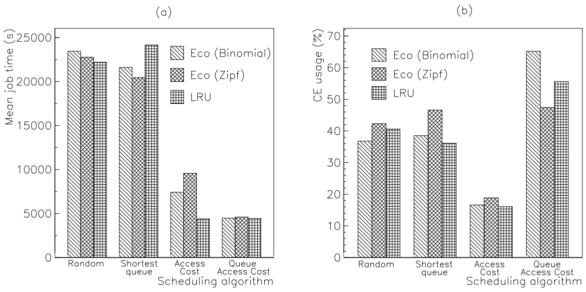

Figure 3. (a) Mean job time and (b) CE usage for various optimisation algorithms and sequential access pattern.

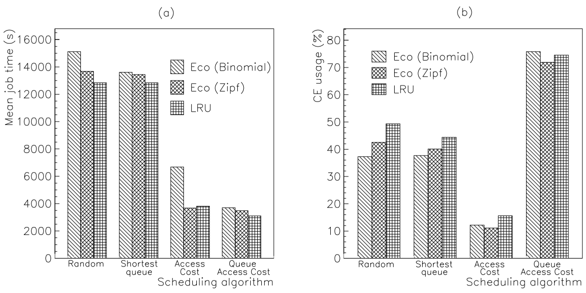

Figure 4. (a) Mean job time and (b) CE usage for various optimisation algorithms and Gaussian random walk access pattern.

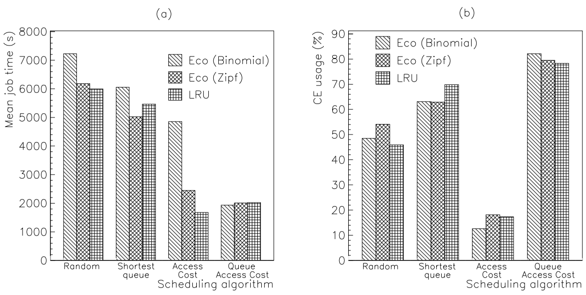

Figure 5. (a) Mean job time and (b) CE usage for various optimisation algorithms and Zipf access pattern.

The SE usage for each of the above scheduling strategies was also monitored as the simulation progressed, considering the three optimisation strategies with sequential access patterns.

Figure 6(a) shows that the *Random* and *Shortest Queue* strategies quickly filled up all the available SEs to reach the maximum of 90%, while *Access Cost* only used the SEs with high network connectivity, resulting in slower execution time and a final SE usage level of only 37%. *Queue Access Cost*, on the other hand, took network connections into account but also avoided long queues of jobs, resulting in good SE usage and fast execution time.

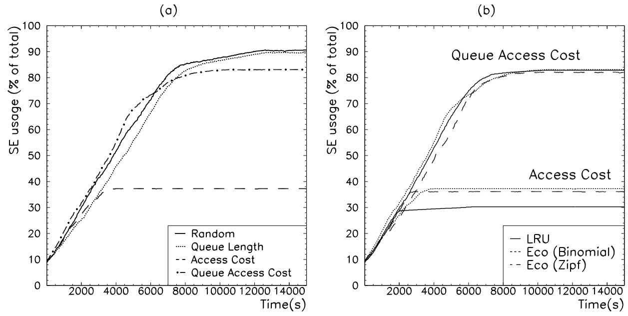

Figure 6. SE usage: (a) All schedulers, binomial economic model (b) All optimisation strategies, *Queue Access Cost* and *Access Cost* schedulers

All optimisation strategies gave very similar results, as can be seen for two of the schedulers in Figure 6(b). Note that 100% SE usage is never reached due to the Grid configuration used, in which CERN and FNAL serve as data repositories to which no jobs are sent.

We therefore use the *Queue Access Cost* scheduling algorithm, with sequential access patterns, for all further tests.

7.2 Scalability of Optimisation Algorithms

In the next set of tests we study the scalability of the optimisation algorithms by varying the number of jobs from 100 to 1000 to 10,000. To set a scale for these tests, the number of jobs required to fill the SEs to $\sim$75% was measured. Using the binomial economic model, sequential access patterns and *Queue Access Cost* scheduler, it was found that this level was reached after $\sim$ 500 jobs had been completed and hence that the above range was reasonable.

The effective network usage, shown in Figure 7(b), decreases with the number of jobs submitted, as we might expect, since the access histories used by the optimisation strategies to make replication decisions take time to build up and stabilise. The economic strategies, though, show much lower usage with an increased number of jobs, with a factor of 3 difference between the binomial based economic model and the LRU strategy. In short, the main advantage of the economic strategies is that they use up considerably less network bandwidth than the LRU strategy.

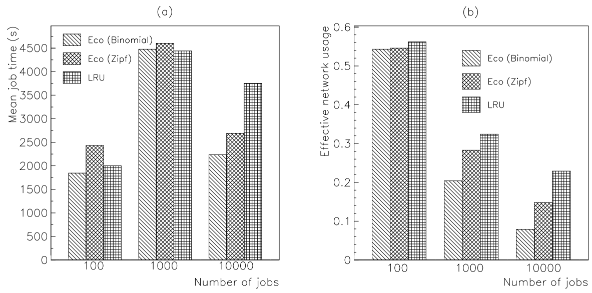

Figure 7. (a) Mean job time and (b) effective network usage for different number of submitted jobs.

This scalability can also be seen in the mean job times (Figure 7(a)), with the economic strategies becoming more effective with an increased job load. From 1000 jobs to 10000 jobs, the mean job time for the binomial based economic strategy fell by 2000s and for the Zipf based economic strategy by 1200s, whereas for the LRU strategy it only fell by 600s.

7.3 Effects of non-Grid Network Traffic

In all the previous evaluations we included non-Grid background traffic in our network model. This non-Grid background traffic consists of any data transfers that can be observed on the network throughout the day; here, we examine the effect this has on Grid performance by comparing results with and without the inclusion of background.

To build up a profile of the underlying network traffic for the testbed links, Iperf [19] monitoring data from various sources was used. For European sites, WP7 of the EDG provide half-hourly monitoring [21]. SLAC and FNAL give two-hourly monitoring of transatlantic links [14], while the UK e-Science Grid Network Monitoring [20] programme gave some data for the UK links, as did the GridNM tool [13]. The sizes of the data samples varied from a period of a few days to about two months. The mean value for each half-hour period of the day was found and a profile of mean available bandwidth as a function of the time of day compiled for each link for which data was available. Where data for a particular link was not available, it was substituted by using data from as similar a link as possible, e.g. a site on the same router.

A selection of these profiles is shown in Figure 8, with the uncertainty in the mean for each point; it can be seen that some links exhibit a clear diurnal variation in the available bandwidth. There is also a large variation in the actual bandwidth available from link to link.

A comparison of the results with and without background traffic is shown in Figure 9. As would be expected, there is a large increase in mean job time when we simulate the background network traffic. For all the optimisation strategies this increases by around a factor of 3.

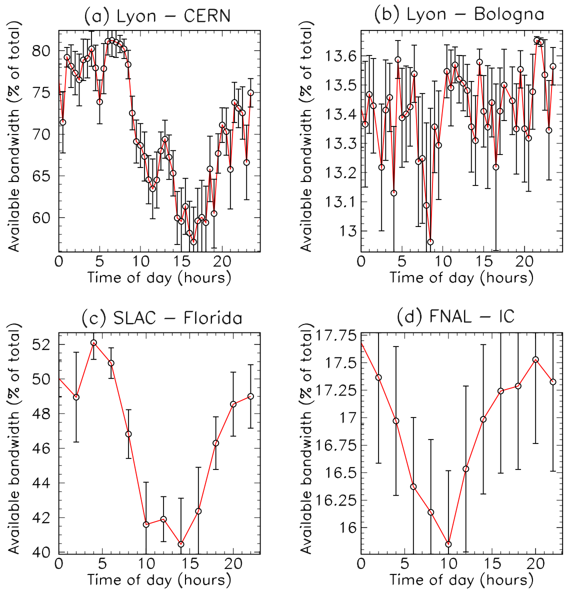

Figure 8. Mean available bandwidth on some of the monitored links.

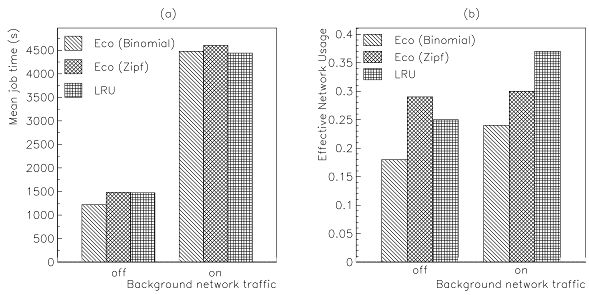

Figure 9. Effects of background network traffic on (a) mean job time and (b) effective network usage

There is also a big increase in the effective network usage for the binomial-based economic strategy and LRU strategy, while for the Zipf-based economic strategy it remains roughly constant. This is perhaps due to the changing network bandwidths leading to less reliable replication decisions by the optimisation strategies, which in the long term means that more replication takes place - except in the Zipf-based economic strategy, which seems to be the most stable to fluctuations.

8 Conclusion

In this paper we have described an environment suitable for the simulation of realistic Grid scenarios and the evaluation of Grid optimisation algorithms. We have discussed various strategies in scheduling and replica optimisation and presented results showing their performance in the tests we have done with the Grid simulator OptorSim.

We have shown that the choice of strategies used can affect the extent to which the Grid resources are exploited and the throughput of Grid jobs. In particular, our experiments show that *Queue Access Cost*, a scheduling algorithm which takes into account both the file access cost of jobs and the workload of computing resources, is the most effective at optimising computing and storage resources and reducing the average time to execute jobs.

We have also proven that a suitable choice of data replication strategy can improve Grid performance; for most situations, particularly with large numbers of jobs, the economy-based strategies we have developed have the greatest effect, regardless of the presence of background (non-Grid) network traffic.

Acknowledgements

This work was partially funded by the European Commission program IST-2000-25182 through the EU DataGrid Project, the ScotGrid Project and PPARC. We would also like to thank James Casey, Heinz Stockinger and Tony Doyle for their feedback on the paper.

References

[1] M. F. Arlitt and C. L. Williamson. Web Server Workload Characterization: The Search for Invariants. In *ACM Sigmetrics Int. Conf. on Measurements and Modeling of Computer Systems*, Philadelphia, PA, USA, May 1996.

[2] P. Barford and M. Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *ACM Sigmetrics Int. Conf. on Measurements and Modeling of Computer Systems*, Madison, WI, USA, July 1998.

[3] W. H. Bell, D. G. Cameron, L. Capozza, P. Millar, K. Stockinger, and F. Zini. Design of a Replica Optimisation Framework. Technical Report DataGrid-02-TED-021215, CERN, Geneva, Switzerland, December 2002.

[4] W. H. Bell, D. G. Cameron, L. Capozza, P. Millar, K. Stockinger, and F. Zini. OptorSim - A Grid Simulator for Studying Dynamic Data Replication Strategies. *Int. Journal of High Performance Computing Applications*, 17(4), 2003.

[5] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, P. Millar, K. Stockinger, and F. Zini. Evaluation of an Economy-Based File Replication Strategy for a Data Grid. In *Int. Workshop on Agent Based Cluster and Grid Computing at CC-Grid2003*, Tokyo, Japan, May 2003. IEEE-CS Press.

[6] L. Breslau et al. Web Caching and Zipf-like Distributions: Evidence and Implications. In *IEEE INFOCOM'99*, New York, NY, USA, March 1999.

[7] R. Buyya and M. Murshed. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing. *The Journal of Concurrency and Computation: Practice and Experience*, pages 1–32, May 2002. Wiley Press.

[8] M. Carman, F. Zini, L. Serafini, and K. Stockinger. Towards an Economy-Based Optimisation of File Access and Replication on a Data Grid. In *Int. Workshop on Agent based Cluster and Grid Computing at CCGrid 2002*, Berlin, Germany, May 2002. IEEE-CS Press.

[9] P. Crosby. EDGSim. `http://www.hep.ucl.ac.uk/~pac/EDGSim/`.

[10] B. T. Huffman et al. The CDF/D0 UK GridPP Project. CDF Internal Note. 5858.

[11] H. Lamehamedi, Z. Shentu, B. Szymanski, and E. Deelman. Simulation of Dynamic Data Replication Strategies in Data Grids. In *Proc. 12th Heterogeneous Computing Workshop (HCW2003)*, Nice, France, April 2003. IEEE-CS Press.

[12] V. Lefebure and T. Wildish (editors). The Spring 2002 DAQ TDR Production. CMS Internal Note. 2005/000, Geneva, Switzerland.

[13] Y-T. Li. GridNM. `http://www.hep.ucl.ac.uk/~ytl/monitoring/gridnm/gridnm-client.html`.

[14] C. Logg et al. SLAC WAN Bandwidth Measuring Tests. `http://www.slac.stanford.edu/comp/net/bandwidth-tests/antonia/html/slac%_wan_bw_tests.html`.

[15] K. Ranganathan and I. Foster. Identifying Dynamic Replication Strategies for a High Performance Data Grid. In *Proc. of the Int. Grid Computing Workshop*, Denver, CO, USA, November 2001.

[16] K. Ranganathan and I. Foster. Decoupling Computation and Data Scheduling in Distributed Data-Intensive Applications. In *Int. Symposium of High Performance Distributed Computing*, Edinburgh, Scotland, July 2002.

[17] WP2 Optimisation Team. OptorSim, a Replica Optimiser Simulator. `http://cern.ch/edg-wp2/optimization/optorsim.html`.

[18] The European DataGrid Project. `http://www.edg.org`.

[19] A. Tirumala et al. Iperf. `http://dast.nlanr.net/Projects/Iperf/`.

[20] UK e-Science Grid Network Monitoring. `http://gridmon.ucs.ed.ac.uk/gridmon/`.

[21] WP7. Network Services. `http://ccwp7.in2p3.fr/`.

[22] G. K. Zipf. *Selected Studies of the Principle of Relative Frequency in Language* . Cambridge, MA.: Harvard University Press, 1932.