OPTORSIM: A SIMULATION TOOL FOR SCHEDULING AND REPLICA OPTIMISATION IN DATA GRIDS

D. G. Cameron, A. P. Millar, C. Nicholson, University of Glasgow, Glasgow G12 8QQ, Scotland
R. Carvajal-Schiaffino, F. Zini, ITC-irst, Via Sommarive 18, 38050 Povo (Trento), Italy
K. Stockinger, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

Abstract

In large-scale grids, the replication of files to different sites and the appropriate scheduling of jobs are both important mechanisms which can reduce access latencies and give improved usage of resources such as network bandwidth, storage and computing power. In the search for an optimal scheduling and replication strategy, the grid simulator OptorSim was developed as part of the European DataGrid project. Simulations of various high energy physics (HEP) grid scenarios have been undertaken using different job scheduling and file replication algorithms, with the emphasis being on physics analysis use-cases. An economy-based strategy has been investigated as well as more traditional methods, with the economic models showing advantages for heavily loaded grids.

INTRODUCTION

The remit of the European DataGrid (EDG) [1] project was to develop an infrastructure that could support the intensive computational and data handling needs of widely distributed scientific communities. In a grid with limited resources, it is important to make the best use of these resources, whether computational, storage or network. It has been shown [3, 4] that data replication - the process of placing copies of files at different sites - is an important mechanism for reducing data access times, while good job scheduling is crucial to ensure effective usage of resources. A useful way of exploring different possible optimisation algorithms is by simulation, and as part of EDG, the grid simulator OptorSim was developed and used to simulate several grid scenarios such as the CMS Data Challenge 2002 testbed [2]. Although the EDG project has now finished, the work on OptorSim has continued, with emphasis on particle physics data grids such as the LHC Computing Grid (LCG). In this paper, an overview of OptorSim's design and implementation is presented, showing its usefulness as a grid simulator both in its current features and in the ease of extensibility to new scheduling and replication algorithms. This is followed by a description of some recent experiments and results.

SIMULATION DESIGN

There are a number of elements which should be included in a grid simulation to achieve a realistic environment. These include: computing resources to which jobs can be sent; storage resources where data can be kept; a scheduler to decide where jobs should be sent; and the network which connects the sites. For a grid with automated file replication, there must also be a component to perform the replica management. It should be easy to investigate different algorithms for both scheduling and replication and to input different topologies and workloads.

Architecture

OptorSim is designed to fulfil the above requirements, with an architecture (Figure 1) based on that of the EDG data management components. In this model, computing and storage resources are represented by *Computing Elements* (CEs) and *Storage Elements* (SEs) respectively, which are organised in *Grid Sites*. CEs run jobs by processing data files, which are stored in the SEs. A *Resource Broker* (RB) controls the scheduling of jobs to Grid Sites. Each site handles its file content with a *Replica Manager* (RM), within which a *Replica Optimiser* (RO) contains the replication algorithm which drives automatic creation and deletion of replicas.

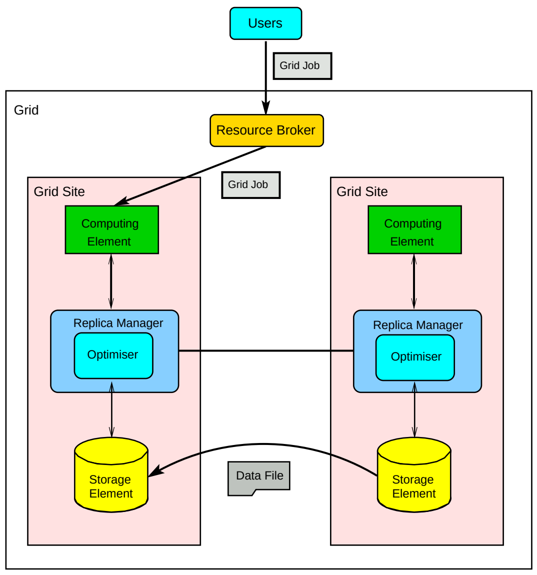

Figure 1: OptorSim Architecture.

Input Parameters

A simulation is set up by means of configuration files: one which defines the grid topology and resources, one the jobs and their associated files, and one the parameters and algorithms to use. The most important parameters include: the access pattern with which the jobs access files; the submission pattern with which the users send jobs to the RB; the level and variability of non-grid traffic present; and the optimisation algorithms to use. A full description of each is in the OptorSim User Guide [5].

Optimisation Algorithms

There are two types of optimisation which may be investigated with OptorSim: the scheduling algorithms used by the RB to allocate jobs, and the replication algorithms used by the RM at each site to decide when and how to replicate.

Scheduling Algorithms. The job scheduling algorithms are based on reducing the "cost" needed to run a job. Those currently implemented are: *Random* (a site is chosen at random); *Access Cost* (cost is the time needed to access all the files needed for the job); *Queue Size* (cost is the number of jobs in the queue at that site); and *Queue Access Cost* (the combined access cost for every job in the queue, plus the current job). Apart from with the *Random* scheduler, the site with the lowest cost is chosen. It should be noted that this "cost" is unrelated to the price of files in the economic model for replication, described below.

Replication Algorithms. There are three broad options for replication strategies in OptorSim. Firstly, one can choose to perform no replication. Secondly, one can use a "traditional" algorithm which, when presented with a file request, always tries to replicate and, if necessary, deletes existing files to do so. Algorithms in this category are the LRU (Least Recently Used), which deletes those files which have been used least recently, and the LFU (Least Frequently Used), which deletes those which have been used least frequently in the recent past. Thirdly, one can use an economic model in which sites "buy" and "sell" files using an auction mechanism, and will only delete files if they are less valuable than the new file. Details of the auction mechanism and file value prediction algorithms can be found in [6]. There are currently two versions of the economic model: the binomial economic model, where file values are predicted by ranking the files in a binomial distribution according to their popularity in the recent past, and the Zipf economic model, where a Zipf-like distribution is used instead.

Implementation

In OptorSim, which is written in Java^{TM}, each CE is represented by a thread, with another thread acting as the RB. The RB sends jobs to the CEs according to the specified scheduling algorithm and the CEs process the jobs by accessing the required files, running one job at a time. In the current implementation, the number of worker nodes for each CE simply reduces the time a file takes for processing, rather than allowing jobs to run simultaneously. When a file is needed, the CE calls the `getBestFile()` method of the RO being used. The replication algorithm is then used to search for the "best" replica to use. Each scheduling and replication algorithm is implemented as a separate Resource Broker or Replica Optimiser class respectively and the appropriate class is instantiated at run-time, making the code easily extensible.

There are two time models implemented, one time-based and one event-driven, and OptorSim can be run in either mode with the same end results. In the time-based model, the simulation proceeds in real time. In the event-driven model, whenever all the CE and RB threads are inactive, the simulation time is advanced to the point when the next thread should be activated. The use of the event-driven model speeds up the running of the simulation considerably, whereas the time-based model may be desirable for demonstration or other purposes.

OptorSim can be run from the command-line or using a graphical user interface (GUI). A number of statistics are gathered as the simulation runs, including total and individual job times, number of replications, local and remote file accesses, volume of storage filled and percentage of time that CEs are active. If using the command-line, these are output at the end of the simulation in a hierarchical way for the whole grid, individual sites and site components. If the GUI is used, these can also be monitored in real time.

EXPERIMENTAL SETUP

Two grid configurations which have been simulated recently are the CMS[1] Data Challenge 2002 testbed (Figure 2) and the LCG August 2004 testbed (Figure 3).

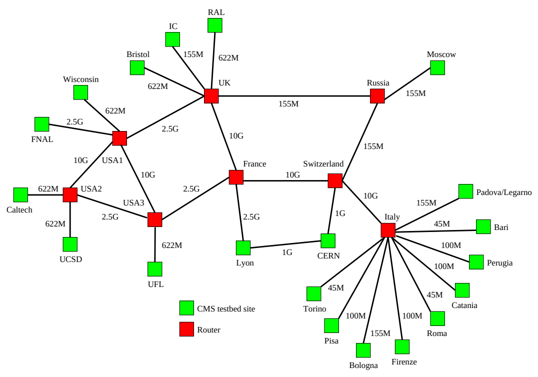

Figure 2: CMS Data Challenge 2002 grid topology.

For the CMS testbed, CERN and FNAL were given SEs of 100 GB capacity and no CEs. All master files were stored at one of these sites. Every other site was given 50 GB of storage and a CE with one worker node. For the LCG testbed, resources were based on those published by the LCG Grid Deployment Board for Quarter 4 of 2004 [7], but with SE capacities reduced by a factor of 100 and number of worker nodes per CE halved, to achieve useful results

[1]Compact Muon Solenoid, one of the experiments for the Large Hadron Collider (LHC) at CERN.

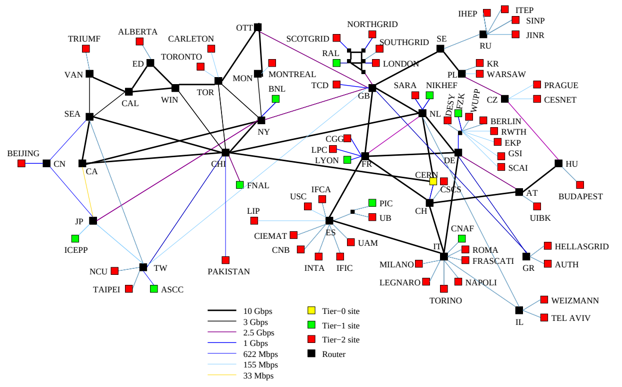

Figure 3: LCG August 2004 grid topology.

while keeping the running time of the simulation low. All master files were placed at CERN. In both cases, a set of 6 job types were run, based on a CDF (Collider Detector at Fermilab) use case [8], with a total dataset size of 97 GB.

Testbed	No. of Sites	$D/\langle SE\rangle$	$\langle WN\rangle$	$\langle C\rangle$ (Mbit/s)
CMS	20	1.764	1	507
LCG	65	0.238	108	463

Table 1: Comparison of Testbeds Used.

In order to compare results from these testbeds, it is necessary to summarise their main characteristics. Useful metrics are: the ratio of the dataset size to the average SE size, $D/\langle SE\rangle$; the average number of worker nodes per CE, $\langle WN\rangle$; and the average connectivity of a site, $\langle C\rangle$. The values of these metrics for the two testbeds are shown in Table 1. Some general statements can be made about these characteristics:

- $D/\langle SE\rangle$. A low value of $D/\langle SE\rangle$ indicates that the SEs have more space than is required by the files. Little deletion will take place and one would expect the different replication algorithms to have little effect.

- $\langle WN\rangle$. A high value of $\langle WN\rangle$ will result in jobs being processed very quickly. If the job processing rate is higher than the submission rate, there will then be little queueing and the mean job time will be short. A low number of worker nodes could lead to processing rate being lower than the submission rate and thus to escalating queues and job times.

- $\langle C\rangle$. A high $\langle C\rangle$ will result in fast file transfer times and hence fast job times. This will have a similar effect on the ratio of job processing rate to submission rate as described above for $\langle WN\rangle$.

Another important factor is the presence or absence of a CE at the site(s) which initially hold(s) all the files. In OptorSim, the intra-site bandwidth is assumed to be infinite, so if a file is local there are no transfer costs involved. For scheduling algorithms which consider the transfer costs, most of the jobs will therefore get sent to that site.

RESULTS

CMS Data Challenge 2002 testbed

With the CMS testbed, three of the replication algorithms (LFU, binomial economic and Zipf-based economic) were compared for the four scheduling algorithms, with 1000 jobs submitted to the grid. The mean job times are shown in Figure 4. This shows that schedulers which

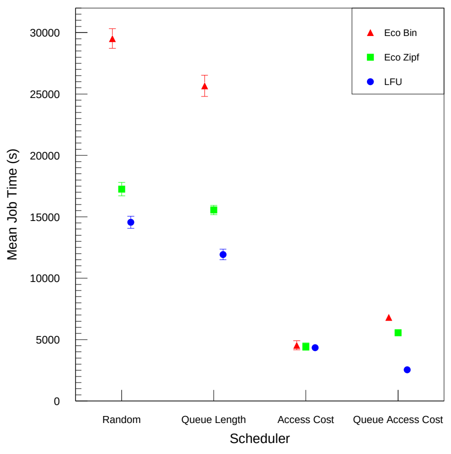

Figure 4: Mean job time for scheduling and replication algorithms in the CMS 2002 testbed.

consider the processing cost of jobs at a site possess a clear advantage, as mean job time is reduced considerably for the *Access Cost* and *Queue Access Cost* schedulers. It can also be seen that the LFU replication algorithm is faster than the economic models for this number of jobs. This may be due to the low value of $\langle WN\rangle$; as the economic models have an overhead due to the auctioning time, there will initially be more queue build-up than with the LFU.

A study was also made of how the replication algorithms reacted to increasing the total number of jobs (Figure 5). As the number of jobs on the grid increases, the mean job time also increases. One would expect that it should decrease if the replication algorithms are effective, but with the low value of $\langle WN\rangle$ in this case, the job submission rate is higher than the processing rate, leading to runaway job times. However, the performance of the economic models improves in comparison to the LFU and when 10000 jobs are run, the Zipf economic model is faster. For long-term optimisation, therefore, the economic models could be better at placing replicas where they will be needed.

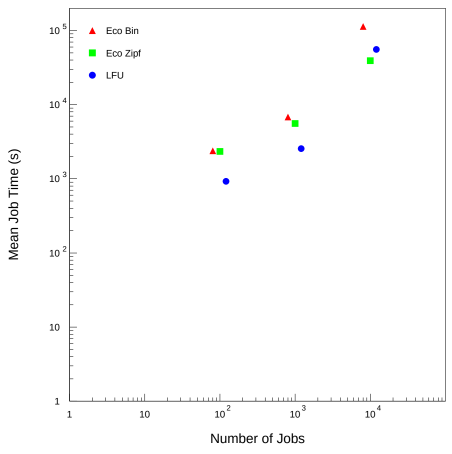

Figure 5: Mean job time for increasing number of jobs in CMS 2002 testbed. Points are displaced for clarity.

LCG August 2004 testbed

The pattern of results for the scheduling algorithms in the LCG testbed (Figure 6) is similar to that for CMS. The *Access Cost* and *Queue Access Cost* algorithms are in this case indistinguishable, however, and the mean job time for the LFU algorithm is negligibly small. This is due to the fact that in this case, CERN (which contains all the master files) has a CE. When a scheduler is considering access costs, CERN will have the lowest cost and the job will be sent there. This is also a grid where the storage resources

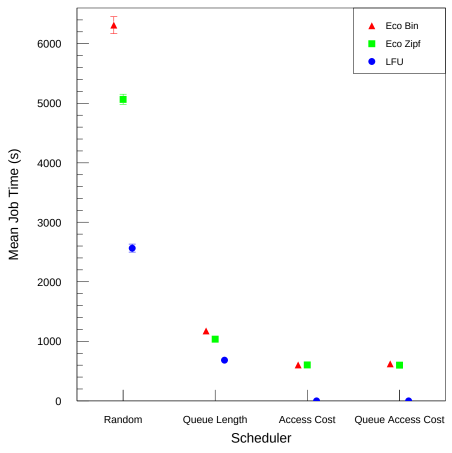

Figure 6: Mean job time for scheduling and replication algorithms in LCG August 2004 testbed.

are such that a file deletion algorithm is unnecessary and a simple algorithm such as the LFU runs faster than the economic models, which are slowed down by the auctioning time. It would therefore be useful to repeat these experiments with a heavier workload, such that $D/\langle SE \rangle$ is large enough to reveal the true performance of the algorithms.

CONCLUSIONS

The investigation of possible replication and scheduling algorithms is important for optimisation of resource usage and job throughput in HEP data grids, and simulation is a useful tool for this. The grid simulator OptorSim has been developed and experiments performed with various grid topologies. These show that schedulers which consider the availability of data at a site give lower job times. For heavily-loaded grids with limited resources, it has been shown that the economic models which have been developed begin to out-perform more traditional algorithms such as the LFU as the number of jobs increases, whereas for grids with an abundance of resources and a lighter load, a simpler algorithm like the LFU may be better.

OptorSim has given valuable results and is easily adaptable to new scenarios and new algorithms. Future work will include continued experimentation with different site policies, job submission patterns and file sizes, in the context of a complex grid such as LCG.

ACKNOWLEDGMENTS

This work was partially funded by the European Commission program IST-2000-25182 through the EU DataGRID Project, the ScotGrid Project and PPARC. The authors would also like to thank Jamie Ferguson for the development of the OptorSim GUI.

REFERENCES

[1] The European DataGrid Project, http://www.edg.org

[2] D. Cameron, R. Carvajal-Schiaffino, P. Millar, C. Nicholson, K. Stockinger and F. Zini, “Evaluating Scheduling and Replica Optimisation Strategies in OptorSim”, Journal of Grid Computing (to appear)

[3] K. Ranganathan and I. Foster, “Identifying Dynamic Replication Strategies for a High Performance Data Grid”, Proc. of the Int. Grid Computing Workshop, Denver, Nov. 2001

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

[5] W. Bell, D. Cameron, R. Carvajal-Schiaffino, P. Millar, C. Nicholson, K. Stockinger and F. Zini, “OptorSim v1.0 Installation and User Guide”, February 2004

[6] W. Bell, D. Cameron, R. Carvajal-Schiaffino, P. Millar, K. Stockinger and F. Zini “Evaluation of an Economy-Based Replication Strategy for a Data Grid”, Int. Workshop on Agent Based Cluster and Grid Computing, Tokyo, 2003

[7] GDB Resource Allocation and Planning, http://lcg-computing-fabric.web.cern.ch/LCG-Computing-Fabric/GDB_reosurce_allocation_planning.htm

[8] B. T. Huffman, R. McNulty, T. Shears, R. St. Denis and D. Waters “The CDF/D0 UK GridPP Project”, CDF Internal Note 5858, 2002