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

David Cameron, University of Glasgow, d.cameron@physics.gla.ac.uk
Ruben Carvajal-Schiaffino, ITC-irst, carvajal@itc.it
Jamie Ferguson, University of Glasgow, fergusjk@dcs.gla.ac.uk
Paul Millar, University of Glasgow, p.millar@physics.gla.ac.uk
Caitriana Nicholson, University of Glasgow, c.nicholson@physics.gla.ac.uk
Kurt Stockinger, Berkeley Lab, Kstockinger@lbl.gov
Floriano Zini, ITC-irst, zini@itc.it

Introduction

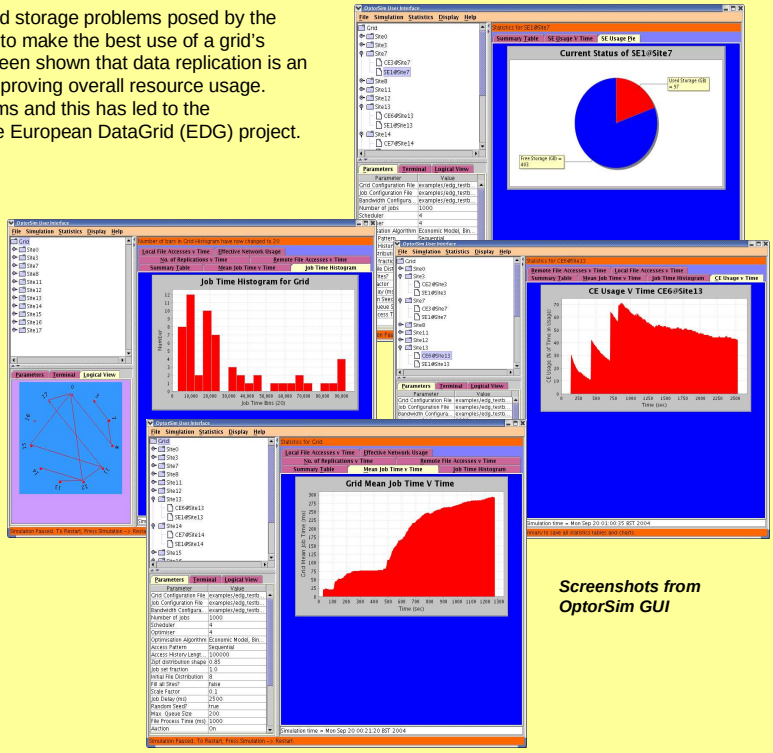
Grid technology is emerging as the solution to the data handling and storage problems posed by the next generation of high energy physics experiments. It is important to make the best use of a grid's resources, whether computational, storage or network, and it has been shown that data replication is an important mechanism for reducing data access times and hence improving overall resource usage. Simulation is a useful way of exploring possible replication algorithms and this has led to the development of the grid simulator OptorSim, originally as part of the European DataGrid (EDG) project.

With OptorSim, it is possible to simulate any grid topology and list of jobs to process by means of a few configuration files. There are several job scheduling and file replication algorithms implemented, and more can easily be added.

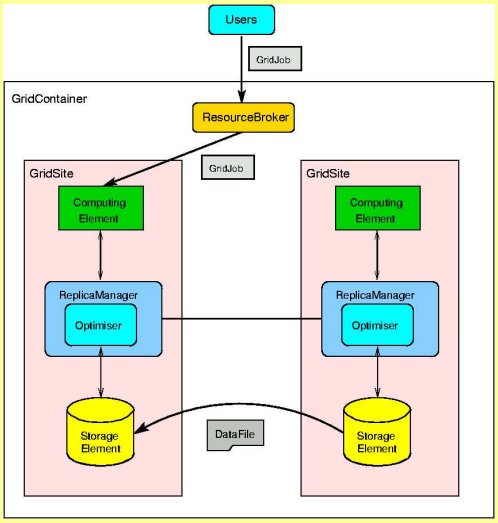
OptorSim can be run from the command line or from a GUI (right). A number of statistics are output:

- Total and individual job times
- CE usage
- Number of replications, local and remote file accesses
- SE usage

The appropriate statistics are output on the level of the grid, individual sites and site components. If the GUI is used, these can also be watched in real time.



Screenshots from OptorSim GUI



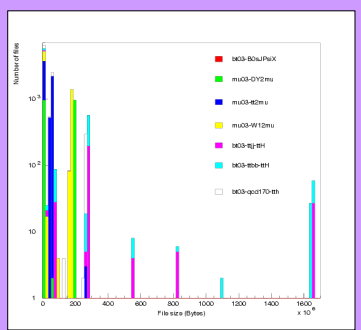
OptorSim architecture

Architecture

OptorSim's architecture (left) is based on that of the EDG data management components. 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. *Users* submit jobs to the grid according to the submission pattern chosen, then a *Resource Broker* (RB) schedules the jobs to Grid Sites. When a job is being processed at a CE, it will go through its list of files to process according to the chosen *access pattern*. If a file is required which is not present on the execution site, it must either be replicated or read remotely. 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.

Inputs

Characterisation of aspects of real grid testbeds such as LCG has been used to get realistic input data, e.g. jobs and files based on the CMS 2004 data challenge (below).



File size distribution from CMS DC04 analysis jobs

Optimisation Algorithms

There are two different types of optimisation which may be investigated using 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 to replicate a file, which file to replicate and which to delete. The overall aim is to reduce the time it takes jobs to run, and also to make the best use of grid resources. In the short term, an individual user wants their job to finish as quickly as possible, but in the long term the goal is to have the data distributed in such a way as to improve job times for all users, thus giving the greatest throughput of jobs. Currently implemented are:

Scheduling

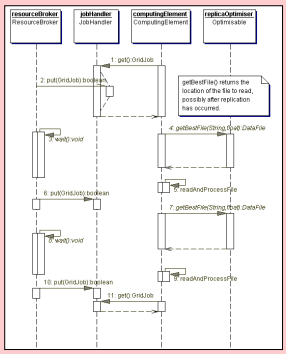
- *Random* - schedule to random site
- *Access Cost* - site where time to access all files required by job is shortest
- *Queue Size* - site where job queue is shortest
- *Queue Access Cost* - site where access cost for all jobs in queue is shortest.

Replication

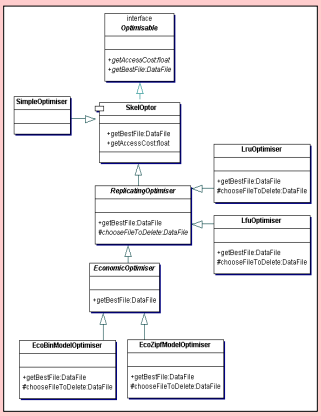
- *No replication*
- *Least Recently Used (LRU)* - always replicate, delete least recently used file
- *Least Frequently Used (LFU)* - always replicate, delete least frequently used file
- *Economic model (Binomial)* - replicate if economically advantageous, using binomial prediction function for file values
- *Economic model (Zipf)* - replicate if economically advantageous, using Zipf-based prediction function.

Implementation

OptorSim is a time-based simulation package written in Java. Each CE is represented by a thread, with another thread acting as the RB and, if the economic model is being used, more threads are used for the auctions and components of the RO which are responsible for them. There are two time models implemented. In *SimpleGridTime*, the simulation proceeds in real time. *AdvancedGridTime* is semi-event driven; when all the CE and RB threads are inactive, simulation time is advanced to the point when the next thread should be activated. 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 until they have finished all their jobs. 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, and the file is either replicated to the local site or read remotely. 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 highly extensible.



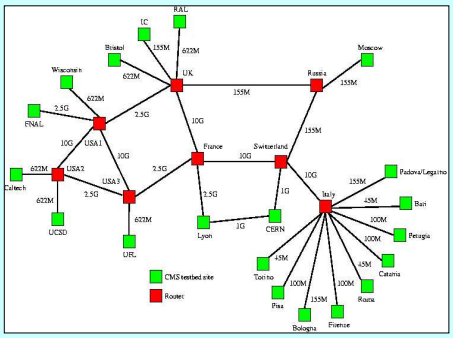
Sequence diagram of CE and RB threads



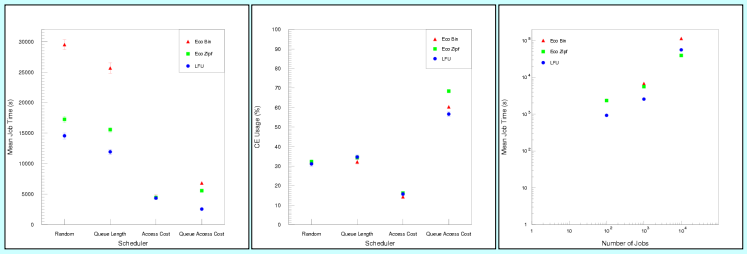
Class diagram of implemented Optimisers

Some Results

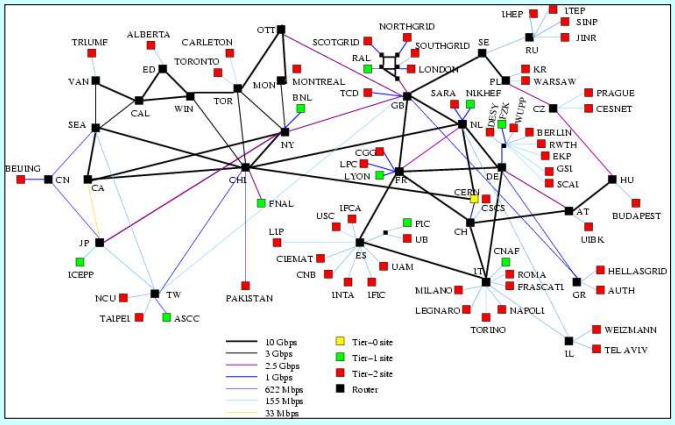
Simulations which have been performed include the EDG testbed, CMS data challenge testbeds and LCG testbed. With the CMS data challenge 2002 testbed (left), it is clear that *Queue Access Cost* is the best scheduler when job times and CE usage is taken into account (below left). The *LFU* replication algorithm is faster when only 1000 jobs are submitted, but the economic models perform better when there are more jobs (below, far right), especially with the Zipf-based prediction function.



CMS DC02 Testbed Topology



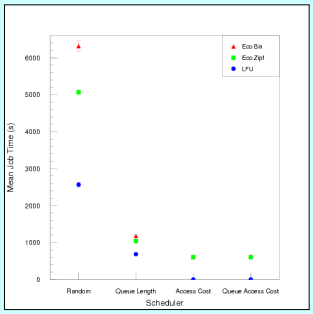
Results from CMS DC02 testbed: (L-R) Mean job time and CE usage for optimisation algorithms with different schedulers, 1000 jobs; mean job time for optimisation algorithms and* Queue Access Cost *scheduler, increasing number of jobs.



LCG August 2004 Testbed Topology

Preliminary results with the LCG testbed of August 2004 (left) show that the *Access Cost* and *Queue Access Cost* schedulers are the fastest (right).

In this case, *LFU* is the fastest of the replication algorithms studied even for a large number of jobs, due to the large SE sizes in this configuration resulting in little replication taking place. Heavier loading of the resources is required to show the true performance of the replication algorithms.



Mean job time for optimisation algorithms with different schedulers, 1000 jobs, LCG testbed

Future Work

Further experimentation with the LCG testbed is ongoing, investigating different site policies, job types and job submission patterns.

OptorSim 2.0 - with new time model, statistics, GUI and job submission options - will be released in the near future.

Further Reading

• *Analysis of Scheduling and Replica Optimisation Strategies for Data Grids Using OptorSim.* D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, C. Nicholson, K. Stockinger and F. Zini. Journal of Grid Computing (to appear).

• *Evaluation of an Economy-Based File Replication Strategy for a Data Grid*. W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini. In International Workshop on Agent based Cluster and Grid Computing at CCGrid 2003, Tokyo, Japan, May 2003

Coming Soon - OptorSim 2.0!

http://edg-wp2.web.cern.ch/edg-wp2/optimization/optorsim.html

GridPP
UK Computing for Particle Physics

ITC irst
CENTRO PER LA RICERCA SCIENTIFICA E TECNOLOGICA

1954-2004 CERN

UNIVERSITY of GLASGOW