

Simulating Resource and Service Trading in Grid Systems

H.A. James and K.A.Hawick
Computer Science Division, School of Informatics
University of Wales, Bangor, North Wales, LL57 1UT, UK
{heath,hawick}@bangor.ac.uk
Tel: +44 1248 38 2717, Fax: +44 1248 36 1429

ABSTRACT

The global grid model is steadily being uptaken by individuals wishing to trade in grid services. The grid was originally driven by organisations attempting to share high performance computing resources but there are recent moves to apply the model to the exchange of other services. It is important that an appropriate economic model be applied to the exchange of services if the global grid approach is to be applied beyond academia and free/share communities. We have built a simulation model of our prototype to investigate the buyer/seller and open market models of services. We are also experimenting with tightly-coupled value-add chains of services in a market economy that would scale to the size of the global grid.

KEY WORDS

Internet; performance modelling; economics and management; grid systems.

1 Introduction

Grid computing [9] is seen to be one of the great hopes for computing in the 21st century. There are many individuals and organisations beginning to adopt so-called Grid Technologies [19, 8, 10] to carry out commercial and research activities that simply were not possible five years ago without dedicated supercomputers.

We are particularly interested in the economic aspects of Grid computing. Until now, most research has focussed on the raw technical aspects of enabling interoperability across widely distributed heterogeneous computing systems. There has been little consideration paid to the economic aspects of such systems; researchers, and to a certain extent, commercial vendors, have either assumed all resources are either *owned* by the same (or cooperating) entities, or that all computing facilities (cycles, storage and attached instrumentation/peripherals) are available *gratis* [20].

We consider the problem of trading resources across the Grid. We structure the knowledge about the Grid as a graph, with the nodes representing physical machines, and the edges representing direct network connectivity between machines. Each machine is considered to offer a number of services (programs) and to also have some capacity to store data relevant to the Grid. Each piece of data or service, has

an associated price, which must be taken into consideration before its use. We also structure the services that are required by client programs as graphs; it is quite possible, especially as the complexity of the graph increases, that the services required to satisfy a client request will not be available on a single machine, therefore some cooperation is required between servers.

Typical Grid simulation research has modelled the Grid as a set of freely-available resources. When modelling Grid systems, the research typically considers a particularly fine-grained approach, modelling individual processors and timeslots on the machines that comprise the Grid environment [4, 6, 18, 1]. In contrast, we take a more high-level view of the Grid system; our macro view considers the trading of services by servers, which may in actual fact utilise uni- or multi-processor machines – the actual target architecture is not as important to this study as the servers and services to be traded. In addition we also consider the case in which services and perhaps data are able to be migrated between machines in order to minimise the amount of network traffic that may be incurred through repeated requests [16].

Initially, our simulation tries to find other available machines connected to the Grid, which are running the appropriate software and are able to service processing (or data) requests. This “resource discovery” phase is enabled by our “gossip protocol” [13], in which each machine keeps a short record of the other machines it knows about, updating the information according to a pre-set policy manager. This process is continuous, and the information that a machine has on the remainder of the system state is used when dealing with requests for processing; our system recognises that it is unlikely a single machine will *ever* have complete system state information and makes allowances accordingly.

When a request for either data retrieval or processing is made to a machine, it then tries to match the query’s requirements against its known list of available Grid resources. A match involves optimisations against time and a notion of ‘price’. In our simulation, machines are able to advertise their services and data at any price; one of the phenomena under investigation is that of machines being able to raise and lower their prices, similar to realistic trading scenarios, as a function of time and of demand. We show that this has the expected effect of creating and re-

ducing the load placed on machines from external request sources.

2 Services on the Grid

The Computational Grid[9] has evolved from the super-computing and high performance computing communities and is largely viewed as a way of sharing resources and particularly costly computing resources. Most typical Grid systems are involved with the sharing of the computing resources for the purposes of scientific computation. As such, when multiple machines are required by jobs, the individual machines must be carefully co-scheduled to allow the work to proceed. Most Grid computing environments (e.g. [8, 10]) are built to enable users to submit their own programs (in the form of source or binary codes) to the system and have it run.

Scientific computation need not be the Grid's main purpose however and we think it highly likely that the grid model will spread and be used for other online services. We have experimented with online image processing and archiving services[14] and found that a data flow or workflow model lends itself well to the organisation of complex service requests on a global scale. We think that a model based on task graphs can be made to scale to global grid sizes. In this paper we describe some scaling models and ideas for determining how services on a global commercial grid might operate.

3 User scenarios and their trading models

In this section we describe some of the user scenarios that we have considered in the development of our different models. We attempt to draw out the differences in the requirements of each model. Finally we describe the type of model that our prototype implementation approximates.

The first user scenario that we describe is based around the commercial problem of trading resources. Companies are divided into producers and consumers. At different stages of the manufacturing processes, companies can be both producers of refined goods and consumers of raw materials. In this scenario we consider that a company will be able to select from whom it purchases raw materials, which should keep costs down. Companies may enter into purchasing agreements with suppliers, guaranteeing business for the supplier, which should also keep costs down. However, if there is not competition for a producer, there is no incentive to reduce costs. This monopolistic behaviour must be tempered by the desire to increase revenue, if not market share.

We consider a second trading scenario that involves the concept of partnerships between companies that are essentially in competition, for example, some airlines. Each airline individually wishes to maximise its market share, but may voluntarily decide to partner with another airline by carrying its overflow passengers or passengers on routes

it does not directly service, in the belief that the same will be true for the other airline, as well as increasing customer loyalty.

The third trading model is the type of model exhibited by large research houses. These research houses will process data of nearly any type, using global research resources, to sell the results of such processing to paying customers. Essentially what they are doing is the computing equivalent of searching, filtering and combining data from various sources to make *value added data*, that is data which is more meaningful than its constituent parts. This approach to problem solving has been used quite successfully in the computer science field for numerical analysis; amongst others, a highly optimised library of mathematical functions has been provided to allow complex problems to be started on personal computers and run on powerful supercomputers [3]. In a modern day setting this is the equivalent of providing a small number of well-known and optimised programs to be executed on a centrally-owned and centrally-managed computer bureau service, which is too low level for most users.

A fourth, and more abstract, type of trading model is that of raw services. We consider the case in which an organisation offers to transparently manage resources for users. For example, in a scientific computing setting, an organisation might offer to manage a user's home directory, providing them with uniform and secure access from anywhere in the world without the need to download files to a local machine. This would make the life of an international scientist a lot simpler. The organisation may also offer to monitor things, such as web sites and data sources, for users, providing them with automatic updates when the events of interest happen. We have chosen to implement this fourth model in our initial study in the knowledge that it may be extended to incorporate features relevant to the first three. The basic model is described in section 4, and the extension of the models characteristics are discussed in section 5.

4 A Basic Model

In this section we describe the initial model that was developed to simulate resource trading in a Grid-sized distributed system. The initial model under consideration was a very simple discrete event simulation built using the *simjava* [17] package developed at the University of Edinburgh. The model consisted of a large number of clients and servers. Each server maintains a list of services that it offers and the price at which the services are traded. Clients create requests, which consist of small strings of services that must be run to satisfy the client's requirements. Thus, we have definite buyers and sellers of resources in what could be classed as a microscopic buyer/seller model, as opposed to an average properties model like the Black-Scholes [2, 7] model.

To simulate the dynamic nature of the global Grid, clients in our initial model, would randomly select a num-

ber of servers from the available list and ask them for a quote to provide the services. Servers would compile a quote, based simply on the sum of their current asking price for each service, and return it to the client for consideration. When the client receives the quotes from the selected servers, a decision is made, on purely economic grounds, which server to use for the request. The client sends the selected server a message requesting it to run the selected services. This is illustrated in figure 1.

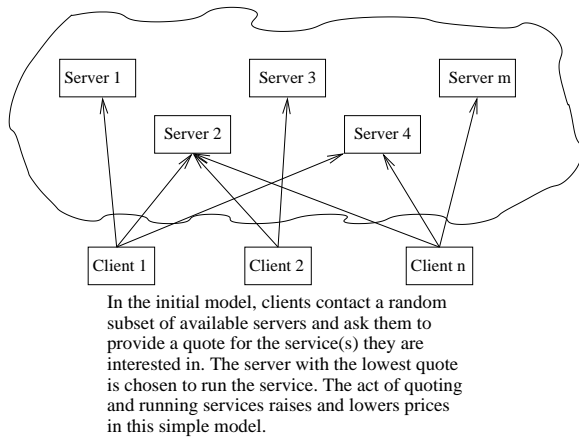


Figure 1. Conceptual view of initial simulation model. Clients choose which servers they will ask for quotes. Not all servers providing a given service will be asked for a quote.

Our initial model was created to illustrate the way in which market forces control the pricing and availability of services in a global Grid environment. As mentioned above, each server maintains a list of services, and the price for which the service may be traded. The price is not fixed; in fact, in our initial simulation the starting price was a random value between an upper and lower bound. Whenever a quote is requested for a service the server has the ability to raise or lower the price depending on the number of unsuccessful quotes it has made in the preceding time quantum. Similarly, when a service is actually traded to a client, the price of the service may be adjusted according to its popularity.

Pseudo-code used to implement the clients and servers is shown in figures 2 and 3. We are currently experimenting with the effects different parameters have on the stability of the system as the number of transactions (and services) approaches the millions. For example, we have variables representing the amount by which individual servers are able to discount their prices when they have been asked to quote on a service many times in between purchases; and correspondingly the amount by which prices should be increased so as to maximise the amount of profit made by a server but at the same time ensuring it maintains its share of the 'market'. We are also studying the amount of overlap of services between Servers and how that may account for some experimental systems

converging upon (or oscillating around) a common price, or diverging rapidly apart.

```
public class Client extends Sim_entity {
    initialise discrete event system
    represent all services by an integer
    get list of available Servers
    public void body() {
        while (running) {
            select a number of random services
            select a random sub-set of Servers
            to ask for a quote
            send a quote request to the
            selected Servers
            wait for quote replies from all
            selected Servers
            calculate the cheapest Server from
            all quotes
            send a buy request to the cheapest
            Server
        }
    }
}
```

Figure 2. Pseudo-code representing the Client in our economic trading model.

In figure 4 we show the fluctuation in service prices over time as produced by our simulation executing the code that implements the pseudo-code above. We see that some services become cheaper due to un-popularity and others become more expensive due to their popularity. The diagram shows the price fluctuation of a single service offered by two servers. Time, in this diagram, is in discrete time steps, and is not representative of wall-clock time. For example, Server1 is initialised in the simulation as having a higher price than Server2. Due to the fact that clients are able to randomly select the servers they want to quote on a particular service, we see the prices of each service diverge for the first half of the graph. Server1's pricing structure becomes more expensive as it is consistently chosen to execute a request by one client that does not know about the existence of Server2. In contrast, Server2's pricing becomes cheaper as it supplies more quotes to clients without being chosen to run the service. From approximately time point 100 onwards we see both Server1 and Server2 being considered by the same client, thus forcing the price of Server1's services downwards and Server2's services upwards. We finally see in the diagram a quasi-equilibrium point being reached at simulation time point 130. After this simulation snapshot we expect to see some minor variations in the pricing of the two Server's services due to adjustments made when one server is chosen over the other to fulfill a service request, and vice versa.

Our preliminary results suggest that the fundamental mechanisms we are modelling should extend to the size of the global Internet. Our models are becoming increasingly complex in nature as we attempt to incorporate the func-

```

public class Server extends Sim_entity {
    initialise discrete event system
    represent all services by an integer
    create a randomly-sized array to hold
        the services offered by this Server
    fill the array with random services
    create a random price for each service
    public void body() {
        receive a message from a client
        if (message is a quote request) {
            if (Server sells requested service(s)) {
                drop the service's price
                send a return message with service(s)
                current selling price
            } else { // Server doesn't sell service
                send a return message without a quote
            }
        } else { // it is a buy request
            if (not first time service has been
                bought by this client) {
                randomly increase price to make
                more profit
            }
        }
    }
}

```

Figure 3. Pseudo-code representing the Server in our economic trading model.

tionality and behaviour of real-life traders in different circumstances. We describe the desired properties of these models and provide some discussion in section 5 and conclude the paper in section 6.

5 Discussion of Trading Models

In this section we describe scaling models and ideas for how services in a global Grid-like environment might operate. We also discuss how our basic model is currently being extended to more realistically model the global Grid. We draw on the discussion of example scenarios in section 3 and discuss some existing approaches and some of the remaining issues for distributed service provision across a wide-area computing environment.

One of the most basic questions is: how are servers discovered and hence selected for quote request from the multitude that are available in a wide-area distributed system the size of the Internet? In our initial simulation, all servers were considered to be known at the time of system start-up. However our more recent models are using a proxy-based system that provides indirect access to the servers. As shown in figure 5, proxies act as well-known servers – in fact the services they provide are ‘service-locating services.’ The proxies collect information about which servers are currently available, and which services each offers. Using proxies allows us to reduce the complex-

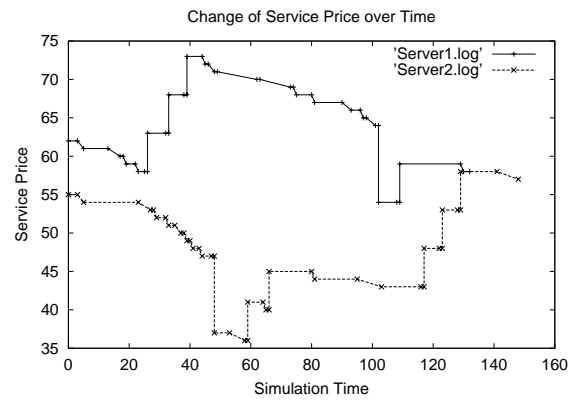


Figure 4. Snapshot of price fluctuations over time. Clients randomly choose which servers they request quotes from, which drives pricing structures up and down.

ity of the client programs and allows users to concentrate on composing meaningful and correct processing queries.

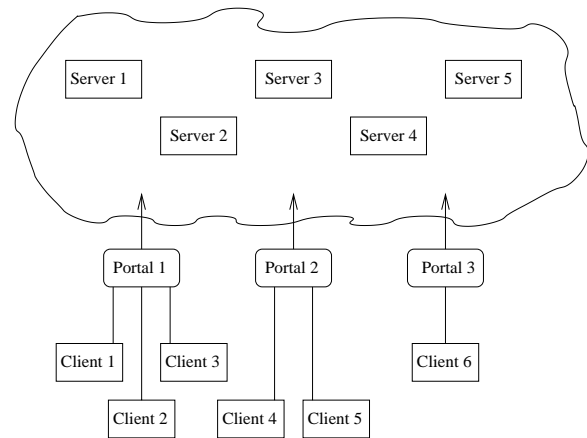


Figure 5. A model for service access on the global grid. Clients request services of portals which allocate the work to servers known to them.

We also consider in our model the method by which our agent chooses the server to provide the required services. In our initial model the method of selection is based simply on the cheapest quote returned by the servers contacted by the client. No consideration is made of any other factors, including: the speed of the service, the reliability of the server for service provision, the distance of the server from the client, or any concept of the amount of data to be sent to (or received from) the service. We are currently investigating the use of user-defined policies to customise the way in which matches are made between client requirements and services on offer.

A feature of the extended model we are currently investigating is a mechanism by which a client is able to negotiate a service price with a server. Our goal is to recreate some real-world scenarios where an initial quote is made

and once a potential server has been identified, a price negotiation phase begins, based on the current price, the client's desired price, and the expected volume of services to be traded through the agreement.

We envisage this trading model as an integral part of our larger Distributed Information Systems Control World [11] (DISCWorld) grid environment. The DISCWorld system allows users to create complex high-level computational jobs by composing together individual service requests in the form of a graph. To be part of the DISCWorld system, a server can offer a set of well-defined services and/or data to the remainder of the distributed environment. Each server maintains its own list of available services and data, and server-specific policy information controls whether requests to copy or manipulate private data are allowed. Servers are allowed to set their own prices for services and data, based primarily upon their usefulness, whether the owner of the resource is happy to have a high load placed on their participating machines (the price of popularity) and the relative speed of the service on their equipment.

Our current system schedules [16] jobs by contacting a known server and asking it to complete the scheduling information for as much of the graph as possible; the presence of alternative resource costings is currently ignored. The schedule is then passed to other services, which can be arranged in a tree-like hierarchy, until the schedule is complete and the execution can begin. The economic model will allow services to be traded between servers and schedules to be devised according to their relative costs and other information which may be considered according to the relative trade-offs involved.

In our extended system incorporated into the larger DISCWorld environment, we suspect we would see a market effect, similar to that shown by our basic model in section 4 and that in real life, of prices raising and lowering subject to the market demand. We expect the system to gradually move towards an equilibrium point where the prices across much of the Grid, or at least in segments, remain relatively static at a coarse granularity. However, when new service providers become available or valuable resources are made available, we suspect this might have a large impact on the stability of the system – similar to a spike in the financial market place today.

Our DISCWorld system also incorporates the ideas of Data Futures, data objects that have not yet been created, but for which a viable schedule has been devised [15, 12]. In our DISCWorld economic model, futures can be represented as stock market options – where the schedule has been pre-determined at a fixed price. The main departure from the financial services model is that when a data future is called upon, the servers that previously agreed to participate in the creation of the resultant data are held to their guarantee.

Some issues that need to be considered:

- Consider a client buying service(s) online that may interlink

- Client needs to trust service source(s)
- Client may need to either download or instigate the service on a remote system
- How is client billed for service?
- How are service providers paid an appropriate share - especially when they have contributed only one part of a value add chain?

6 Conclusions

We approach the problem of economics in Grid systems through the use of a discrete event simulation. Our simulation model is being developed using a publicly-available Java-based toolkit, simjava [17], which has been used in other applications for grid simulations [5]. In this paper we report on preliminary design and experimentation, and provide some discussion on related issues.

We are experimenting with various commercial and economic trading models that can be applied to online services. Internet and WAN technologies are only just making this sort of approach feasible but we expect improvements in security and middleware technologies will create a demand for a well developed economic trading model for online services. We are considering the scalability implications for trading on a global scale.

Our preliminary results suggest that this trading approach is indeed feasible for simulating interactions across a wide-area Grid system, of the size likely to arise over the next few years.

Acknowledgements

The Distributed and High Performance Computing Group is a collaborative venture between the University of Wales, Bangor and the University of Adelaide in South Australia.

References

- [1] K. Aida, A. Takefusa, H. Nakada, S. Matsuoka, S. Sekiguchi, and U. Nagashima, "Performance evaluation model for scheduling in a global computing system", *Int J High Performance Computing Applications*, Vol. 14, No. 3, Sage Publications, USA, 2000.
- [2] F. Black and M. Scholes, "The pricing of options and corporate liabilities", *J. Political Economy* **81**(1973), pp. 637–659.
- [3] S. Browne, J. Dongarra, E. Grosse and T. Rowan, "The Netlib Mathematical Software Repository", in *D-Lib Magazine*, September 1995.
- [4] Rajkumar Buyya, "Economic-based Distributed Resource Management and Scheduling for Grid Computing", PhD Thesis, Monash University, Melbourne, Australia, April 12, 2002 (submitted)

- [5] Rajkumar Buyya and Manzur 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 (CCPE)*, Wiley Press, USA, May 2002.
- [6] H. Casanova, "Simgrid: A Toolkit for the Simulation of Application Scheduling", *Proc First IEEE/ACM Int Symp on Cluster Computing and the Grid (CC-Grid 2001)*, May 2001, Brisbane, Australia.
- [7] N.A. Chriss, "Black-Scholes and Beyond Option Pricing Models", McGraw-Hill, ISBN 0-7863-1025-1, 1997.
- [8] Globus. Available at <http://www.globus.org>
- [9] Ian Foster and Carl Kesselman, eds, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, Inc., 1999. ISBN 1-55860-475-8.
- [10] Avaki Corporation. Available from <http://www.avaki.com>
- [11] K.A. Hawick, H.A. James, A.J. Silis, D.A. Grove, K.E. Kerry, J.A. Mathew, P. D. Coddington, C.J. Patten, J.F. Hercus, and F.A. Vaughan, "DISCWorld: An Environment for Service-Based Metacomputing," *Future Generation Computing Systems (FGCS)*, 15:623–635, 1999.
- [12] K.A. Hawick and H.A. James, "Data Futures in Metacomputing Systems", DHPC Technical Report DHPC-075. Available from <http://www.dhpc.informatics.bangor.ac.uk/reports>. November 1999.
- [13] K.A. Hawick and H.A. James, "Modelling a Gossip Protocol for Resource Discovery in Distributed Systems", in *Proc. Parallel and Distributed Processing Techniques and Algorithms (PDPTA)*, Las Vegas, June 2001.
- [14] K.A. Hawick, P.D. Coddington, H.A. James and C.J. Patten, "On-Line Data Archives", in *Proc. Hawaii Int. Conf. on Systems Sciences (HICSS 34)*, Maui 2001.
- [15] H.A. James and K.A. Hawick, "Remote Application Scheduling on Metacomputing Systems," *Proc. HPDC'99*, Redondo Beach, California, August 1999.
- [16] Heath A. James. "Scheduling in Metacomputing Systems", PhD Thesis, The University of Adelaide, 1999.
- [17] R. McNab and F.W. Howell, "Using Java for Discrete Event Simulation" in *proc. Twelfth UK Computer and Telecommunications Performance Engineering Workshop (UKPEW)*, Univ. of Edinburgh, 219-228, 1996.
- [18] H. Song, X. Liu, D. Jakobsen, R. Bhagwan, X. Zhang, K. Taura, and A. Chien, "The MicroGrid: A Scientific Tool for Modeling Computational Grids", *Proc IEEE Supercomputing (SC 2000)*, Nov. 4-10, 2000, Dallas, USA.
- [19] Sun Microsystems. Sun Grid Engine. Available from <http://www.sun.com>
- [20] M. Thompson-Lawrence. "An Investigation into a Computational Grid Economy". BSc Honours Thesis, University of Wales, Bangor. June 2002.