A Simulation Framework for Studying Economic Resource Management in Grids

Kurt Vanmechelen, Wim Depoorter, and Jan Broeckhove

University of Antwerp, BE-2020 Antwerp, Belgium
kurt.vanmechelen@ua.ac.be

Abstract. Economic principles are increasingly being regarded as a way to address conflicting user requirements, to improve the effectiveness of grid resource management systems, and to deliver incentives for providers to join virtual organizations. Because economic resource management mechanisms can encourage grid participants to reveal the true valuations of their jobs and resources, the system becomes capable of making better scheduling decisions. A lot of exploratory research into different market mechanisms for grids is ongoing. Since it is impractical to conduct analysis of novel mechanisms on operational grids, most of this research is being carried out using simulation. This paper presents the Grid Economics Simulator (GES) in support of such research. The key design goals of the framework are enabling a wide variety of economic and non-economic forms of resource management while simultaneously supporting distributed execution of simulations and exhibiting good scalability properties.

1 Introduction

Conducting research into resource management systems (RMS) on real grids is difficult for two main reasons. The first one relates to the costs involved in setting up and maintaining such a system. The second is the need to test new RMS's under a variety of different load patterns and infrastructural arrangements, which is all but impossible to achieve with a real grid system. The large scale on which grid RMS's need to be studied exacerbates these problems. The only viable option for researchers then is to resort to simulation. While there exists a number of general purpose simulators for grids, they have limited support for economic resource management systems (ERMS). There is a need for such support however, as it allows easy *comparison* between different economic and non-economic approaches and enables researchers to *focus on the mechanism design and implementation* of the chosen approach, while leveraging the strength of the existing general purpose framework in *setting up* the grid environment, *running* the simulation and *monitoring* the desired metrics.

M. Bubak et al. (Eds.): ICCS 2008, Part I, LNCS 5101, pp. 226–235, 2008.

2 Related Work

To provide some background on existing simulators and their capabilities we describe a number of them [1,2,3,4,5,6,7] here. For a more elaborate overview one can consult [8].

The Bricks simulator was designed as a performance evaluation system to analyse different scheduling approaches for High Performance Computing systems in a global setting [1]. Two of the most interesting features of Bricks are the use of a *scripting language* to describe the configuration and parameters of the simulation and its ability to *incorporate external components* such as NWS into simulations. Bricks has also been used to evaluate fixed cost-based scheduling approaches [9]. The framework dictates a centralized approach for resource management however, limiting its general applicability. Development has ceased and the framework is no longer available from the official project site.

MicroGrid can create *virtual Globus environments* of arbitrary composition and allows for the execution of real applications [2]. As such, it is actually an *emulator* rather than a simulator. This makes MicroGrid interesting for optimizing grid applications with regards to the target configuration of the grid or conversely allow designers of grids to play with various parameters to optimize the grid architecture. Since MicroGrid is an emulator running real applications, it is very time intensive. It is also difficult to test new resource management approaches as all of them have to be compatible with Globus. Active development seems to have halted after 2004.

SimGrid [3] is an extensive toolkit for the simulation of distributed applications and is written in C. The toolkit started out with a central scheduling approach and was subsequently adapted to allow for decentralized scheduling [10]. Later on, it was extended in order to allow developers to implement distributed services in the simulator and transfer them to a real world grid without code modification. Development is ongoing with the addition of MPI support and modifications to the networking layer. SimGrid focuses heavily on the network aspects of grids and less on scheduling strategies. To accommodate for economic resource management, substantial modifications would have to be made to make the simulated entities economic aware and to support the required interaction patterns. While SimGrid has been used in combination with economic scheduling approaches [11], the auctions were performed outside of the framework, with SimGrid only executing the resulting schedule.

GridSim is written in Java on top of the SimJava 2.0 basic discrete event infrastructure, dating from 2002. GridSim allows for packet-level simulation of the network and also offers components oriented towards data grids. Additionally, it supports advance reservations, workload traces, an output statistics framework and background network traffic. GridSim has been used to simulate a Nimrod-G like deadline and budget constrained scheduling system [4] and an auction environment [5]. Development is ongoing with the latest release dating from September 2007.

OptorSim [6] is a discrete event simulator that has been developed to simulate data access optimization algorithms in grids. In this regard, it takes inter-site

bandwidth into account for data transfers between grid sites. The simulator's focus is on overall optimization of grid resources rather than intra-site or per-user optimization. This allows OptorSim to simplify two aspects; all users are modeled as a single *Users* entity and the worker nodes at each grid site are represented by a single entity as well. The simulation model is based on a simplification of the architecture proposed by the European DataGrid (EDG). OptorSim has been used to evaluate cost-based replication-aware algorithms for Resource Brokers and Replica Optimization Services (ROS). The latest version was released in October 2006.

jCase [7] is a tool for evaluating combinatorial auctions through simulation. It has been applied to the field of grid resource management and supports multiple algorithms for price determination and solvers for determining the optimal set of winners in a combinatorial auction. As such, it is one of the few simulation tools that support research into ERMS's. jCase however, is not a general purpose framework and specifically targets combinatorial auctions. Currently it also lacks support for simulations of dynamic systems over time.

3 GES Overview

The Grid Economics Simulator (GES) is a discrete event simulator that has been developed for the evaluation of various economic approaches in their ability to efficiently organize a resource market. This section will present an overview of the simulator's architecture, operation and features.

3.1 Key Abstractions

Since the focus of GES is on economic grid resource management, we will describe the key abstractions from an economic point of view. It is important to note however that GES also supports non-economic resource management in which case aspects such as billing and pricing are omitted.

The *consumer* represents a grid user that wants to execute computational *jobs*. Each consumer has a queue of jobs that need to be executed and for which resources must be acquired from *providers* through participation in the market. A consumer is provided with a budgetary endowment that may be replenished periodically. In every simulation step, consumers are billed with the usage rate prices for all resources that are allocated to their jobs at that particular moment.

Every provider hosts a number of CPU and disk resources that are supplied to the computational market. Providers interact with consumers to agree upon a price for the execution of a job. When agreement is reached, the provider will bill the consumer. The execution of a job may start immediately or in the future. Once a resource is allocated to a job, it remains allocated until the job completes.

The *market* brings together consumers and providers. It also dictates the interaction pattern used for negotiating resource allocations. A market has a *bank* facility that keeps accounts for each consumer and provider. The bank also handles all transactions necessary for paying the bills associated with resource usage.

A market follows either a spot market or future market allocation paradigm. The former is characterized by immediate dispatching of a job to a resource while the latter supports advance reservation. A more in depth explanation on these allocation paradigms is given in 3.5.

3.2 Simulation Parameters

All simulated entities are characterized by a number of parameters. The most important ones relate to the number of consumers and providers participating in the simulation, the number of jobs and their induced workload, the budgets of the consumers, and the number of CPUs and their collective processing capacity. For increased flexibility, values for these parameters may be chosen in a multitude of ways and at different grouping levels as supported by the `configuration` layer described in the next subsection.

For example, the average number of jobs in the simulation N^{aj} is related to the normalized total system load L and the average normalized load of a job l^{aj} by $L = N^{\mathrm{aj}} \times l^{\mathrm{aj}}$. Therefore it is possible to choose two of these three parameters to fully specify the load of a scenario. When we want to simulate the arrival of jobs over time we can also use traces of job arrivals T^{j} or approximate them using arrival distributions D^{j}.

The previous discourse is also applicable on a consumer group level as well as on the individual consumer level. For consumer group i for example, we can choose values for the average number of jobs in the group N_i^{aj} and the average normalized load of a job l_i^{aj}. The translation of these averages into concrete values for each consumer can be done in a straightforward way by distributing them equally over all consumers in the group, but also by means of a chosen random distribution.

3.3 Architecture

An overview of the GES architecture is given in the layered diagram of figure 1. Each layer is mapped to a package in the simulator's codebase.

One of the key design goals of the architecture are extensibility and reusability. This "extend-and-refine" philosophy can be found throughout the whole `simulation core` layer and its components. The `domain` layer contains base classes for all domain entities such as `Consumer`, `Provider`, `Job`, `GridResource` and `GridEnvironment`. The `Bank` entity is situated in the `economic` layer. Support for traditional forms of resource management is provided through the `non-economic` layer. Class extension is heavily used from the domain layer up to the specific RMS implementation. For instance, a `Consumer` class of the `Domain` layer only keeps track of job status metrics, while an `EconomicConsumer` also keeps track of budgetary metrics. Existing components can be easily extended when new RMS algorithms are added to the framework. An overview of the different RMS systems that are currently supported by GES is given in section 3.5.

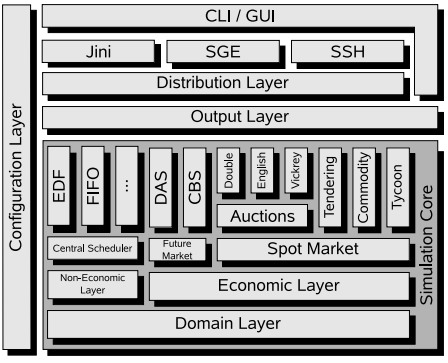

Fig. 1. Overview of the architecture of GES

Examples of reusability can be found in the `Economic` layer which provides components for accounting, billing and transactions, the `Future Market` layer which hosts reservation mechanisms for preemptible and non-preemptible workloads, the `Auctions` layer that supports pluggable protocols for auctioning, and the `Tendering` layer where new negotiation strategies can be plugged in.

Simulations can be distributed over multiple processing nodes through the `distribution` layer. This layer interfaces with compute resources that host a Jini-enabled compute service, clusters fronted by a Sun Grid Engine head node, or clusters with a passwordless SSH setup. Currently, distribution is supported at the granularity of a simulated scenario. Possibilities for distributed execution of the individual entities in the simulation are planned for future releases.

The `gui` layer allows the user to create, run and monitor live market scenarios. A screenshot of the user interface is given in figure 2. A persistency framework

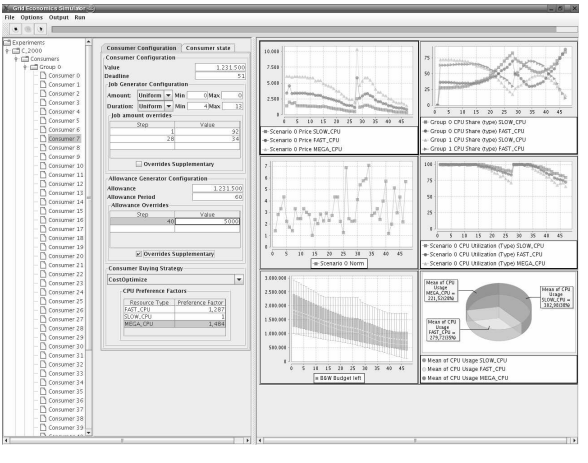

Fig. 2. Screenshot of the GES UI

allows for storing both scenario configurations and configurations of the UI layout. Aggregated metrics over simulation runs and over a selection of simulated entities (e.g. a collection of consumers) are supported in the form of means, variances, standard deviations and box plots. After data collection and analysis, data can be directly exported from the simulator's UI to standard data formats such as `csv` or graphical formats such as `eps` and `png`.

3.4 Operational Overview

A simulation runs for a number of time steps. Each time step consists of a number of phases. For the spot markets (see 3.5), these phases are listed on figure 3. First a central controller updates the joblist and budget of the consumer. Then, depending on the market mechanism used, the consumers, providers or both are instructed to start negotiations. In order to execute jobs, the consumer accepts a bill and sends it to the bank in phase 4. In phase 5 all monetary transactions take place. Finally providers are instructed to execute the relevant jobs. When these are finished, the consumer is notified in phase 7.

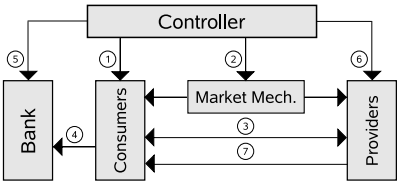

Fig. 3. Overview of a simulation step in GES

3.5 RMS Frameworks

GES comes with built-in support for a number of reference and experimental resource management systems, both non-economic and economic. The non-economic RMS's are provided as a reference and for the purpose of comparison. We have implemented an offline central scheduler that can be initialized with different non-economic scheduling policies:

- An `Earliest Deadline First` policy that schedules in the jobs of the consumer with the earliest deadline first.
- A `Priority` policy where jobs are processed in order of the consumer's configured priority level.
- A `Round Robin` policy scheduling jobs from different consumers in a round robin fashion.
- A `FIFO` policy that schedules jobs in first-in-first-out manner as they arrive.
- The `DONE` policy that aims to maximize the number of consumers that meet their deadline. It follows a greedy approach, scheduling in consumer requests in order of increasing workload. When planning in an individual job, the CPU with the largest available remaining processing capacity is selected and the job is planned in as close as possible to its deadline.

The economic RMS's implemented in GES are divided into two separate branches. The first one encompasses the spot markets while the second one incorporates the future markets. Spot markets are characterized by very dynamic price setting and quick reaction to changing conditions but also suffer from the exposure problem [12]. Future markets with support for advance reservation and co-allocation solve this problem at the expense of increased complexity and reaction time to changing market conditions. In spot markets, consumers have to negotiate per job for execution rights, while in future markets they can do this for an entire application consisting of multiple jobs. The spot markets that are implemented in GES are the following:

- A `Selective Tendering` market with congestion control [13], where consumers request quotes from a group of selected providers. If a consumer is unable to obtain an allocation after requesting a certain number of quotes, it backs off and tries again at a later point in time.
- An `Auction` market which supports double auctions as well as English, Dutch, First-Price Sealed-Bid and Vickrey auctions [14].
- A `Commodity` market that uses a Walrasian Auctioneer [15] for pricing. Multiple price adjustment schemes can be used ranging from a routine based on Smale's method [16] to various optimization routines delivered by the Matlab Optimization Toolbox which are interfaced through RMI.
- An implementation of the market mechanisms used in `Tycoon` [17].

The future markets supported by GES are:

- The `CBS` [18], a centralized brokering system where consumers have to direct their application processing requests to a central broker entity that will negotiate with the providers. The broker aims to maximize the total value generated by fulfilling the consumers' requests.
- The `DAS` market [18] is a decentralized auctioning system where each provider holds auctions for selling its resources over the scheduling window. A consumer will place a sealed bid for each of its jobs at potentially multiple providers. These providers then calculate the winners of the auctions using a greedy heuristic. Multiple rounds can be held in order to schedule in as much consumers as possible.

4 Case Study: Value Realization for Users with Hard Deadlines

In this case study, we will use GES to study the difference in value realization between different RMS's for consumers which assign a hard deadline to the execution of their application. We compare the economic `DAS` and `CBS` markets with a non-economic, deadline-based scheduler that adopts the `DONE` policy.

We varied the processing capacity of the Grid while measuring realized value, infrastructure utilization, price levels and resource shares. For each sample point, we requested 100 runs in order to monitor the variance in the output metrics as a

result of the use of stochastic variables. In total, 5700 simulations were necessary for the data collection. The simulation was run for 2016 simulated time steps with a grid environment hosting 300 consumers and 20 providers. Consumers were divided into three groups with different deadline ranges and associated valuation factors ($VF_{deadline}$) as shown in figure 4 (left). Every consumer hosted between 210 and 390 jobs with each job having a processing requirement between 1 and 80 time steps. Each consumer's valuation was determined by multiplying a base valuation of 10000 credits with a load dependent factor and the $VF_{deadline}$ factor. For this setup, we assumed consumers to bid truthfully and consequently equated each consumer's bid with its valuation. The total processing capacity in the system was 1250, which was uniformly distributed over the providers in the environment. The processing capacity of each individual CPU varied between 0.5 and 1.

We ran our simulations on the CalcUA cluster at the university of Antwerp which hosts 256 Opteron 250 nodes using GES' distribution layer. Our experiment took 10069 seconds on the cluster, yielding a speedup of 85. This speedup closely corresponded to the amount of nodes available to us on the cluster.

The right graph in figure 4 shows the percentagewise increase of realized consumer value compared to the `DONE` RMS when varying infrastructural capacity. As can be observed from the graph, both the CBS and DAS markets compare favourably to the non-economic approach.

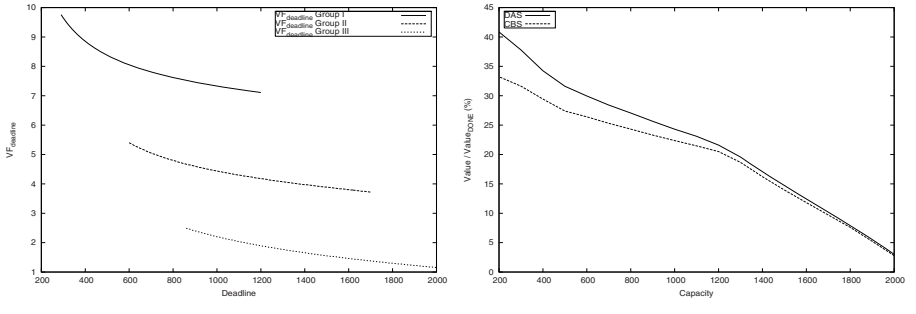

Fig. 4. Valuation factors for the three consumer groups (left) and the value increase for the `CBS` and `DAS` markets compared to the `DONE` RMS, under varying capacity (right)

Table 1 and 2 show the results for the `CBS` market and `DONE` RMS respectively. Standard deviation is shown only for the utilization and value metrics due to space considerations. Although the non-economic approach attains higher utilization levels as a consequence of its preference for smaller workloads, it does not realize as much value for users as the economic approach. The high-value consumer groups are allotted a greater share in the `CBS` market because of their larger budgetary endowment and valuations. In the non-economic approach, the low-value group is given the largest share because it has the execution window with the least amount of resource competition. Table 1 shows that cost levels

Table 1. Output metrics under varying capacity for the `CBS` market

$Cap.$	$Util.(\%)$	$Value(\%)$	$Share_{\mathrm{I}}(\%)$	$Share_{\mathrm{II}}(\%)$	$Share_{\mathrm{III}}(\%)$	$Cost_{\mathrm{I}}$	$Cost_{\mathrm{II}}$	$Cost_{\mathrm{III}}$
2000	79.28±1.41	95.54±0.78	37.23	37.60	25.17	1.90	1.59	0.85
1000	81.82±1.08	58.82±0.90	56.96	28.07	14.97	6.25	2.87	0.89
200	83.45±1.40	12.59±0.23	56.72	29.20	14.08	7.02	3.35	1.05

Table 2. Output metrics under varying capacity for the `DONE` RMS

$Cap.$	$Util.(\%)$	$Value(\%)$	$Share_{\mathrm{I}}(\%)$	$Share_{\mathrm{II}}(\%)$	$Share_{\mathrm{III}}(\%)$
2000	83.99±1.30	92.94±2.12	32.29	33.34	34.37
1000	92.40±1.51	48.06±1.62	29.68	31.49	38.84
200	93.08±1.67	9.45±0.69	29.16	29.58	41.26

per unit of workload adjust to the degree of congestion in the system and the budgetary capabilities of the different consumer groups.

5 Summary and Future Work

Economic forms of resource management offer great opportunities for building grids that deliver incentives for provider participation and that try to maximize realized consumer value. There is a need for general purpose simulators with economic support to assist research in this field. We have introduced the Grid Economics Simulator and illustrated its extensibility by describing its architecture and operation and by providing an overview of the different supported RMS's. We demonstrated the capabilities of GES with a case study highlighting various aspects of the framework.

While GES in its current form has proven to be very useful in our research [15,18,19], we are planning for the inclusion of additional features. The first is the inclusion of *network abstractions*. This is a necessary step for more realistic simulation and to enable planned future research towards bandwidth pricing. In addition, we wish to be able to import traces from *workload databases* such as the Grid Workloads Archive [20]. This would allow us to use more realistic user and job profiles in simulations.

References

1. Takefusa, A., Matsuoka, S., Nakada, H., Aida, K., Nagashima, U.: Overview of a performance evaluation system for global computing scheduling algorithms. In: Proceedings of HPDC 1999, pp. 97–104. IEEE Computer Society, Los Alamitos (1999)
2. Song, H.J., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K., Chien, A.: The microgrid: a scientific tool for modeling computational grids. Sci. Program. 8(3), 127–141 (2000)

3. Casanova, H.: Simgrid: a toolkit for the simulation of application scheduling. In: Proceedings of CCGrid 2001, pp. 430–437. IEEE Computer Society, Los Alamitos (2001)
4. Buyya, R.: Economic-based Distributed Resource Management and Scheduling for Grid Computing. PhD thesis, Monash University, Australia (2002)
5. Assunção, M.A., Buyya, R.: An evaluation of communication demand of auction protocols in grid environments. In: Proceedings of GECON 2006, pp. 24–33. World Scientific, Singapore (2006)
6. Cameron, D.G., Millar, A.P., Nicholson, C., Carvajal-Schiaffino, R., Stockinger, K., Zini, F.: Analysis of Scheduling and Replica Optimisation Strategies for Data Grids Using OptorSim. Journal of Grid Computing 2(1), 57–69
7. Schnizler, B.: Resource Allocation in the Grid; A Market Engineering Approach. PhD thesis, University of Karlsruhe (2007)
8. Sulistio, A., Yeo, C.S., Buyya, R.: A taxonomy of computer-based simulations and its mapping to parallel and distributed systems simulation tools. Softw. Pract. Exper. 34, 653–673 (2004)
9. Takefusa, A., Casanova, H.: A study of deadline scheduling for client-server systems on the computational grid. In: Proceedings of HPDC 2001, pp. 406–415 (2001)
10. Legrand, A., Lerouge, J.: Metasimgrid: Towards realistic scheduling simulation of distributed applications. Technical Report 2002-28, LIP (2002)
11. Das, A., Grosu, D.: Combinatorial auction-based protocols for resource allocation in grids. In: Proceedings of PDSEC 2005, IEEE Computer Society, Los Alamitos (2005)
12. Bykowsky, M.M., Cull, R.J., Ledyard, J.O.: Mutually destructive bidding: The fcc auction design problem. Journal of Regulatory Economics 17(3), 205–228 (2000)
13. Depoorter, W.: Establishment of agency as an effective market based resource allocation method using ges. Master's thesis, University of Antwerp (2007)
14. Vanmechelen, K., Broeckhove, J.: A comparative analysis of single-unit vickrey auctions and commodity markets for realizing grid economies with dynamic pricing. In: Veit, D.J., Altmann, J. (eds.) GECON 2007. LNCS, vol. 4685, pp. 98–111. Springer, Heidelberg (2007)
15. Stuer, G., Vanmechelen, K., Broeckhove, J.: A commodity market algorithm for pricing substitutable grid resources. Fut. Gen. Comput. Syst. 23(5), 688–701 (2007)
16. Smale, S.: A convergent process of price adjustment and global newton methods. Journal of Mathematical Economics 3(2), 107–120 (1976)
17. Feldman, M., Lai, K., Zhang, L.: A price-anticipating resource allocation mechanism for distributed shared clusters. In: Proceedings of EC 2005, British Columbia, ACM Press, New York (2005)
18. Vanmechelen, K., Depoorter, W., Broeckhove, J.: Economic grid resource management for CPU bound applications with hard deadlines. In: Proceedings of CCGrid 2008, IEEE Computer Society, Los Alamitos (in press, 2008)
19. Vanmechelen, K., Stuer, G., Broeckhove, J.: Pricing substitutable grid resources using commodity market models. In: Proceedings of GECON 2006, pp. 103–112. World Scientific, Singapore (2006)
20. Iosup, A., Li, H., Jan, M., Anoep, S., Dumitrescu, C., Wolters, L., Epema, D.H.J.: The Grid Workloads Archive. FGCS (submitted, 2007)