GES 2007
German e-Science

Available online at http://www.ges2007.de

Pricing Resources in Dynamic Grid Economies

K. Abdelkader[1], and J. Broeckhove[1]

Department of Mathematics and Computer Sciences, University of Antwerp,
Middelheimlaan 1, 2020, Antwerp, Belgium
email: {Khalid.Abdelkader, Jan.Broechove}@ua.ac.be
phone: (+32 03) 265 3214, *fax:* (+32 03) 265 3204

Abstract

We investigate pricing of resources in dynamic grids that are modelled as computational commodity markets. Price determination based on achieving supply-and-demand equilibrium has been shown to work for static grids. However some grids are dynamic in the sense that the number of systems or users participating in the grid fluctuates significantly over time. We show that the pricing scheme does not suffer from the disruptive effects of the dynamic grid fabric or user population.

1 Introduction

To generate broad support for Grids, but also to develop usage models that are attuned to the user's needs, it is important that there be a shift from system-centric resource management to a user-centric approach. The focus should be on allocation algorithms that are driven by the user's valuation of the task to be executed. This way Grids will deliver the maximum utility to individual users.

A promising approach towards this goal, involves using an economics based resource manager [1] which takes resource utilization cost into consideration. Such an economics based trading model, where consumers rent resources from providers, is an attractive method to manage resource allocation in Grid systems.

The introduction of economic models in Grid resource management involves the definition of a computational market in which Grid users and providers interact. While most market models have been based on auctions, commodity market models have also received some attention [6]. In this contribution we consider the commodity market model. It corresponds to a vision of the Grid where applications can treat computational and storage resources as interchangeable, and not as specific machines and disk systems.

Obviously, prices are a key element of economics based resource management. They reflect the users' valuations of the resource usage and signal the abundance or scarcity of various types of resources. In the price scheme that we adopt, market participants express their valuation of the resources as a function of its price. That means that for each price level the consumers, users wanting to obtain resources to have their jobs executed, indicate how many resources they will acquire. Similarly, for each price level the providers, organizations or individuals bringing their resources into the Grid, indicate how many of their resources they will make available. The market sets the actual price by computing the price at which supply equals demand. The algorithm that we use for computing this equilibrium is based on Smale's method [2], a well known method from quantitative economy, augmented and extended in a number of ways [5].

This pricing scheme has been investigated with modelling and simulation for Grids with a static configuration i.e. both providers of resources in the grid fabric and the consumers, users having jobs executed on the resources, are fixed in time [5]. Under those circumstances the determination of the equilibrium price under variations of e.g. job load or disposable budget is fast and stable.

For Grids that are based on a dynamic fabric such as desktop Grids or systems based on volunteer computing or on cycle stealing, resource providers and consumers engage and disengage with the Grid in a dynamic fashion. Thus we need to verify whether the pricing scheme can still function well in these circumstances. We need to ascertain whether the changes, sometimes abrupt, in the number of resources does not disrupt the equilibrium algorithm. And, given the variations in the number users and the ensuing total job load, we need to ascertain whether the pricing scheme can handle mismatches between job volume and available resource volume i.e. situations of under- and overdemand.

2 Commodity Market model

The computational market model has been implemented in a Java based discrete event simulator. It enables us to study differing factors such as market size, resource categorizations, budget allocations and participant strategies, and in the present study, the effect of the Grid being a dynamic system. Modeling a resource market involves the resources that are traded, the pricing mechanism, and the behavior of the market participants, the "providers" and "consumers".

2.1 Resources

In this contribution, we limit ourselves to a single type of resources, namely CPU's. In order to introduce diversification related to the CPU performance, we introduces two categories: $fastCPU$ and $slowCPU$. The former have a performance ratio of two compared to the latter, i.e. the $fastCPUs$ execute jobs twice as fast as the $slowCPUs$. These categories constitute substitutable commodities, as jobs can execute on both, although they will be valued differently by consumers.

We use the term resource type to distinguish e.g. CPU, disk, network i.e. non-substitutable resources. The term resource category refers to a partition within a resource types based e.g. on performance. The latter are substitutable, in that a task may execute on both a slow or fast CPU but the user will of course have a preference. In this contribution we only consider one type and two categories.

2.2 Consumers

Each consumer has a queue of CPU-bound computational jobs that need to be executed and for which resources must be acquired from providers. The price a consumer is willing to pay depends on the valuation of the particular job.

The dispatch of a job to the CPU is effected immediately after the necessary resource has been acquired. Initially, every consumer's queue has a number of jobs in it; with a certain probability jobs are added to the queue at every time step. Every job has a nominal running time T [5], i.e. the time it takes to finish on a reference CPU. However, in our algorithms we do not assume that the consumer has knowledge of this running time.

Each consumer is provided with an initial budget and an additional periodic budget allowance. In every simulation step, consumers are charged with the usage rate prices [5] for all grid resources that are currently allocated to their jobs. Consumers do not attempt to save up credits, but try to use their entire budget. However, expenditures are spread evenly across the allowance period. This is done because we assume that consumers do not have reliable estimates of running time of their jobs. Therefore, we need to prevent consumers from agreeing to a price, a "cost" level that would not be sustainable for them over the entire allowance period.

Consumers formulate their demand in the market by expressing their willingness to buy resources, given a price vector P. The components of that vector P_i represent the price per resource unit, per time unit of the i^{th} commodity category that is characterized by $PerformanceRatio_i$ in relation to the reference CPU. Depending on the job mix a consumer has to schedule, certain resource categories will be preferred over others. This is expressed through the $ValuationFactor_i$. In all this leads to an adjusted price for each category, given by

$$AdjustedPrice_i = P_i/(PerformanceRatio_i * ValuationFactor_i) \quad (1)$$

The r.h.s reflects the price normalized to unit performance and factors in the valuation. The consumer expresses demand, limited by the currently remaining budget and the allowed rate of expenditure, in the category with the lowest adjusted price.

The use of the $ValuationFactor_i$ in the adjusted price is a simple abstraction for the complex logic a consumer might follow to prefer one CPU category over another. An example of such a logic whereby a consumer is willing to pay more than double the price for a CPU of category 1, which is only twice as fast as one of category 2, is the following. Suppose the consumer has a job graph that includes a critical path and that a scheduling strategy is used to optimize for total turnaround time. Such a consumer would be willing to pay more than the nominal worth of a CPU of category 2 for allocating jobs on the critical path, as they have a potentially large effect on turnaround time.

2.3 Providers

Every provider hosts a number of CPUs in either category that can be supplied to the computational market. Once a resource is allocated to a job, it remains allocated until the job completes. Also, the market price at the time the resource is sold, will be charged as a fixed rate to the consumer for the duration of the job. This approach is consistent with the fact that we do not assume a prior knowledge of a job's running time.

For a given price vector, providers have to determine how many CPUs they are willing to sell for each CPU category. To do so a quantity MPR_i is calculated. It is the "mean provider revenue per time step and per resource" for category i i.e. total revenue divided by the time step and the number of resources of that category for that provider. The MPR_i reflects the price the provider was able to obtain, on average, in the past for that category.

Given the price vector P_i and a number of resources PC_i, the provider indicate a willingness to supply a number of $Spupply_i$ resources, given by

$$Supply_i = PC_i * \min(1.0, \frac{P_i}{MPR_i}) \tag{2}$$

In other words, at a price that exceeds average past revenue, then all resources are made available; at a price below that level, a share proportional to the ratio is made available. Providers thus limit their supply to the market in order to keep prices high, thereby trying to maximize revenue instead of maximizing utilization by supplying all resources all of the time, regardless of revenue.

However, the fewer resources are sold, the lower the MPR_i becomes, and this will in turn increase the number of resources offered at price level P_i. The duration of the history period used to determine the MPR_i has a significant impact on the speed with which the provider reacts to market circumstances. At one extreme, when the window reduces to the current time step, the $MPR_i \rightarrow P_i$ and the providers makes all resources available. at the other extreme, when the window includes all previous time steps, the MPR_i becomes rigid and short term evolution has little impact. The length of the window can be used to encode the provider's reluctance to react to short term change.

2.4 Market pricing

The price of resources is dynamic, depending on the supply and demand at each time step. That is, the market adjusts the price to bring about market clearing viz. a matching of supply and demand. Prices play the role of a communicator of complex provider and consumer valuations of different resource types and services [5, 6].

The market operates by requesting all parties to provide their supply or demand for a range of prices for each CPU category. This information is used, after pre-processing and smoothing, to define an excess demand surface i.e. the difference between current demand and supply as a function of the price vector. The market equilibrium point is the zero of this surface and fixes the price at which the market will trade at that point in time. The global zero search algorithm is based on Smale's method [3] and the intricacies of the pre-processing, a.o. related to the fact that we are dealing with discrete resource units, can be found in [4, 5].

2.5 Dynamics of resources and providers

In this contribution we have used the term "grid" in a broad sense as a large network of independent provider and consumer nodes cooperating to achieve computational tasks. We interpret this as an inherently dynamic setup, with nodes leaving and joining or rejoining the grid, for whatever reason.

The dynamic aspect of the grid is modelled by having a pool of potential providers and a pool of active providers. At each time step, a random number drawing is effected for each active provider. If this number falls below a pre-set threshold, the provider exits the market and joins the pool of potential providers. Similarly, with a different pre-set threshold, potential providers can become active providers. The size of the pools, and the balance of the thresholds determines the evolution in the number of active providers.

In the current implementation we have not looked at the problem of resubmitting jobs that are lost due to a provider exiting. We simply de-activate the provider and let the provider's CPU's finish processing the jobs currently executing on it. Only then does the provider enter the pool of potential providers.

An analogous mechanism is in place for consumers. Active consumers have jobs and participate in the market. There is a pool of potential consumers that can enter the market. State changes are determined by comparing randomly drawn numbers against pre-set thresholds. When a consumer is de-activated and joins the pool of potential consumers, the jobs in its queue are kept there.

number of fastCPU's per provider	$\{1, 2, \cdots, 20\}$
number of slowCPU's per provider	$\{2, 3, \cdots, 20\}$
performance ratio of fast vs slow	2.0
job running time in time steps	$\{2, 3, \cdots, 10\}$
initial number of jobs per consumer	$\{1, 2, \cdots, 150\}$
probability of new jobs per time step	15%
job injection period in time steps	50
jobs submitted at injection step	$\{1, 2, \cdots, 150\}$
initial budget	500 000
recurrent allowance period time steps	50

Table 1: Simulation parameters that apply to all scenarios.

3 Results

Our investigation focuses on the effect of the dynamic character of the grid. We verify that the market operates correctly, i.e. the equilibrium price can be determined at each time step, and we investigate the price evolution and the ensuing allocation efficiency in a number of scenarios.

Obviously, a simulation involves many parameters: some apply to the model as a whole, others to individual consumers or providers, still others to individual jobs or resources. The key parameters that apply in all scenarios discussed in this section, are listed in table 1. When a range is indicated, the parameter is determined by random drawing in that range.

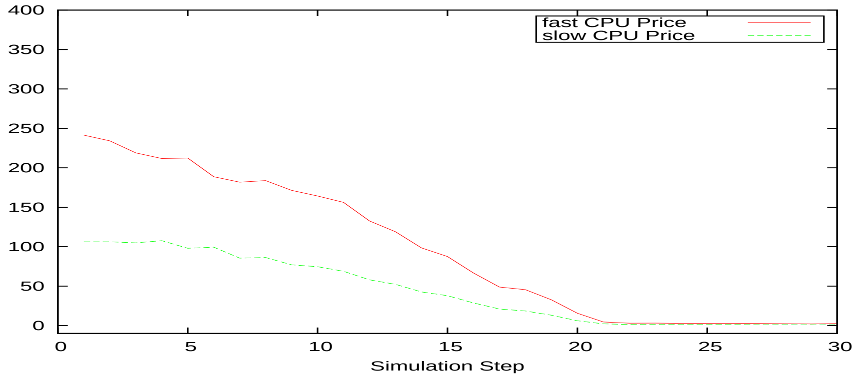

Figure 1: Short term CPU price evolution for the static scenario.

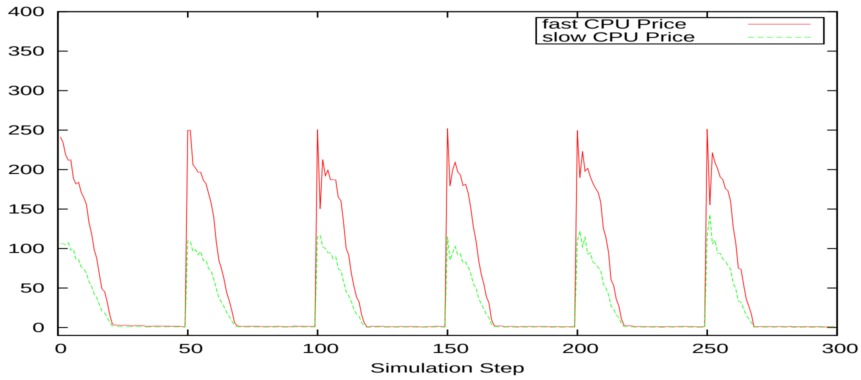

Figure 2: Long term CPU price evolution for the static scenario.

3.1 Static grid scenario

In order to create a point of comparison, we first make a calculation in which the dynamic features have been suppressed. We take 100 active consumers and 50 active providers and those remain active in the market at all times. The initial behavior of the prices in shown in figure 1.

In order to probe the stability of the pricing mechanism we periodically inject a large number jobs into the market and gauge the market's response in terms of price levels, as shown in figure 2. It confirms correct behavior: each time a large number of jobs are injected the price rises and then gradually returns to zero when all jobs are processed.

initial size of the active consumer pool	100
threshold of the active consumer pool	0.1
initial size of the potential consumer pool	100
threshold of the potential consumer pool	0.1
initial size of the active provider pool	50
threshold of the active provider pool	0.1
initial size of the potential provider pool	50
threshold of the potential provider pool	0.1

Table 2: Simulation parameters for the dynamic scenario.

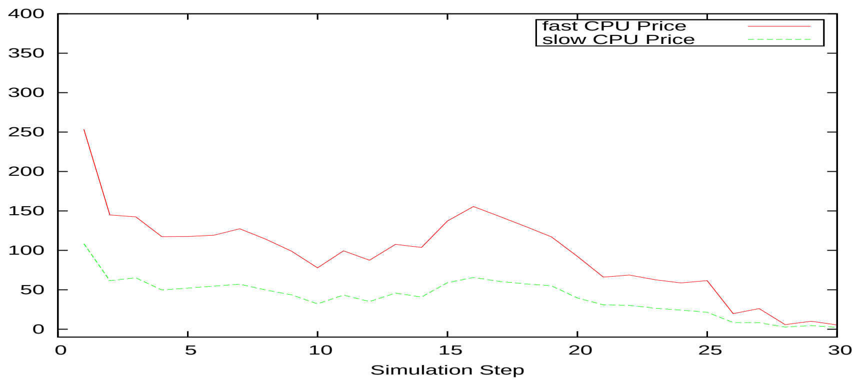

Figure 3: Short term price evolution for the dynamic scenario.

3.2 Dynamic grid scenario

The simulation parameters that apply to this scenario are presented in table 2. They represent a situation in which at each time step ten percent of the

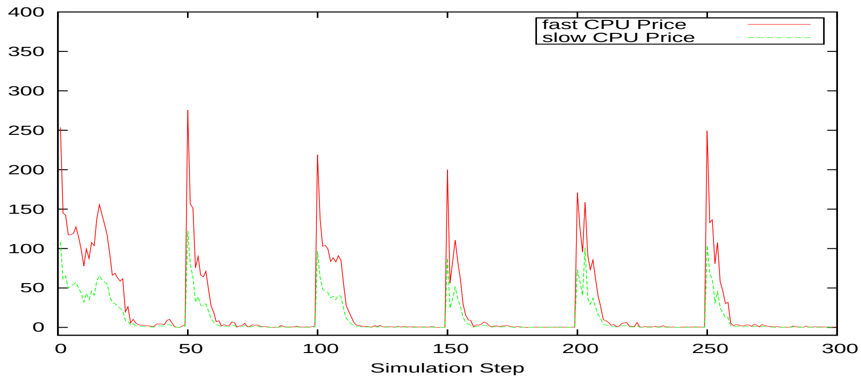

Figure 4: Long term prices evolution for the dynamic scenario.

consumers switch from active to non-active status and vice-versa. The same threshold has been used with the providers.

In figure 3 we show the short term price evolution in this scenario. Clearly, it is more erratic than in the static scenario but that simply reflects the nature of the situation. The key issue is that the pricing scheme still succeeds in determining equilibrium prices.

For the long term price evolution, we again periodically inject large numbers of jobs into the system. The price evolution is shown in figure 4. It is more erratic than in the similar static scenario, but essentially the same observation holds as before: the pricing scheme operates correctly.

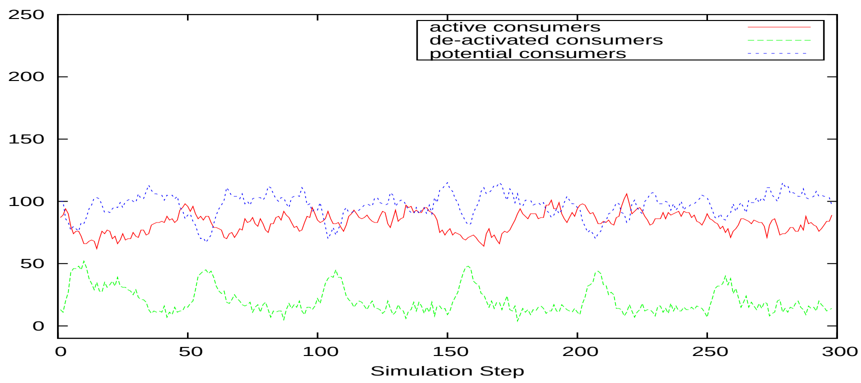

Figure 5: Evolution of the number of consumers in each of the possible states.

In figure 5 we show the time evolution of the number of consumers in the active, de-activated and potential pools. Because both consumer threshold are

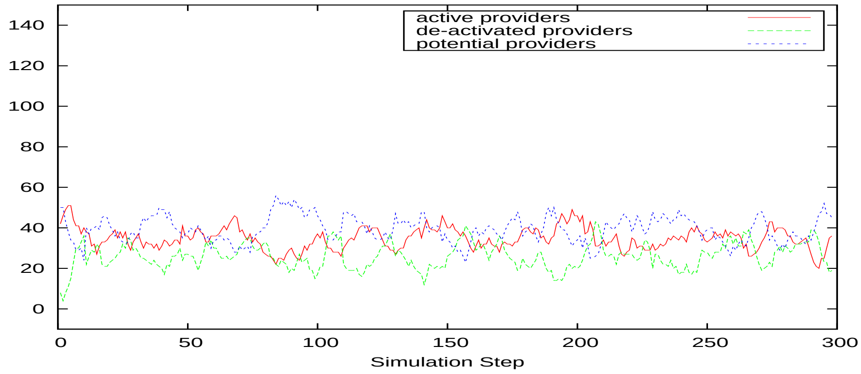

Figure 6: Evolution of the number of providers in each of the possible states.

equal the flow back and forth between both pools is equal on average. The same situation holds in figure 6. for the providers. The relative change in the numbers are significant for both, indicating the robustness of the pricing algorithm.

4 Future work

Of course, this is a first study of the applicability of a commodity market-based pricing scheme to a dynamic grid. More works need to be done with larger numbers of providers and consumers. The computational time of the simulation scales roughly linearly so we definitely intend to pursue that aspect of the investigation. Also the sensitiveness of the result to an number of parameters such as the job length, the threshold levels, and so on needs to be investigated more thoroughly.

5 Conclusion

We have investigated the pricing of resources in dynamic grids modelled as computational commodity markets. We have shown that a scheme based on supply-and-demand equilibrium can work in that context.

References

[1] R. Buyya, D. Abramson, and J. Giddy. Economic models for resource management and scheduling in grid computing. *Concurrency and Computation: Practice and Experience (14)*, pages 1507–1542, 2002.

[2] M.W. Hirsch and S. Smale. Algorithms for solving f(x)=0. *Communications on Pure and Applied Mathematics*, 32:281–312, 1979.

[3] Stephen Smale. Dynamics in general equilibrium theory. *The American Economic Review*, 66(2):288–294, May 1976.

[4] G. Stuer, K. Vanmechelen, and J. Broeckhove. A commodity market algorithm for pricing substitutable grid resources. *Future Generation Computer Systems*, In Press:doi:10.1016/j.future.2006.11.004, 2007.

[5] K. Vanmechelen, G. Stuer, and J. Broeckhove. Pricing substitutable grid resources using commodity market models. In *Proc. of the 3th Int. Workshop on Grid Economics and Business Models*, pages 103–112. World Scientific, Singapore, 2006.

[6] R. Wolski, J.S. Plank, J. Brevik, and T.Bryan. Analysing market-based resource allocation strategies for the computational grid. *Int. J. of High-Performance Computing Applications, 15(3)*, 1:15(3), 2001.