Analyzing Market-based Resource Allocation Strategies for the Computational Grid*

Rich Wolski† James S. Plank† John Brevik‡ Todd Bryan†

† Department of Computer Science
University of Tennessee

‡ Mathematics and Computer Science Department
College of the Holy Cross

Abstract

In this paper, we investigate G-commerce — computational economies for controlling resource allocation in Computational Grid settings. We define hypothetical resource consumers (representing users and Grid-aware applications) and resource producers (representing resource owners who "sell" their resources to the Grid). We then measure the efficiency of resource allocation under two different market conditions: commodities markets and auctions. We compare both market strategies in terms of price stability, market equilibrium, consumer efficiency, and producer efficiency. Our results indicate that commodities markets are a better choice for controlling Grid resources than previously defined auction strategies.

1 Introduction

With the proliferation of the Internet comes the possibility of aggregating vast collections of computers into large-scale computational platforms. A new computing paradigm known as the Computational Grid [17, 3] articulates a vision of distributed computing in which applications "plug" into a "power grid" of computational resources when they execute, dynamically drawing what they need from the global supply. While a great deal of research concerning the software mechanisms that will be necessary to bring Computational Grids to fruition is underway [3, 16, 20, 8, 4, 24, 21, 1, 34], little work has focused on the resource control policies that are likely to succeed. In particular, almost all Grid resource allocation and scheduling research espouses one of two paradigms: centralized omnipotent resource control [18, 20, 28, 29] or localized application control [9, 4, 2, 19]. The first is certainly not a scalable solution and the second can lead to unstable resource assignments as "Grid-aware" applications adapt to compete for resources.

In this paper, we investigate **G-commerce** — the problem of dynamic resource allocation on the Grid in terms of computational *market economies* in which applications must buy the resources they use from resource suppliers using an agreed-upon currency. Framing the resource allocation prob-

*This work was supported in part by NSF grants EIA-9975020, EIA-9975015, and ACI-9876895.

lem in economic terms is attractive for several reasons. First, resource usage is not free. While burgeoning Grid systems are willing to make resources readily available to early developers as a way of cultivating a user community, resource cost eventually must be considered if the Grid is to become pervasive. Second, the dynamics of Grid performance response are, as of yet, difficult to model. Application schedulers can make resource acquisition decisions at machine speeds in response to the perceived effects of contention. As resource load fluctuates, applications can adjust their resource usage, forming a feedback control loop with a potentially non-linear response. By formulating Grid resource usage in market terms, we are able to draw upon a large body of analytical research from the field of economics and apply it to the understanding of emergent Grid behavior. Last, if resource owners are to be convinced to federate their resources to the Grid, they must be able to account for the relative costs and benefits of doing so. Any market formulation carries with it an inherent notion of relative worth which can be used to quantify the cost-to-benefit ratio for both Grid users and stake-holders.

While there are a number of different plausible G-commerce market formulations for the Grid, we focus on two broad categories: **commodities markets** and **auctions**. The overall goal of the Computational Grid is to allow applications to treat computational, network, and storage resources as individual and interchangeable commodities, and not specific machines, networks, and disk or tape systems. Modeling the Grid as a commodities market is thus a natural choice. On the other hand, auctions require little in the way of global price information, and they are easy to implement in a distributed setting. Both types of economies have been studied as strategies for distributed resource brokering [11, 35, 25, 6, 7, 10]. Our goal is to enhance our deeper understanding of how these economies will fare as resource brokering mechanisms for Computational Grids.

To investigate Computational Grid settings and G-commerce resource allocation strategies, we evaluate commodities markets and auctions with respect to four criteria:

1. Grid-wide price stability
2. Market equilibrium
3. Application efficiency
4. Resource efficiency

Price stability is critical to ensure scheduling stability. If the price fluctuates wildly, application and resource schedulers that base their decisions on the state of the economy will follow suit, leading to poor performance, and therefore ineffectiveness of the Grid as a computational infrastructure. Equilibrium measures the degree to which prices are fair. If the overall market cannot be brought into equilibrium, the relative expense or worth of a particular transaction cannot be trusted, and again the Grid is not doing its job. Application efficiency measures how effective the Grid is as a computational platform. Resource efficiency measures how well the Grid manages its resources. Poor application and/or resource efficiency will mean that the Grid is not succeeding as a computational infrastructure. Thus, we use these four criteria to evaluate how well each G-commerce economy works as the basis for resource allocation in Computational Grids.

The remainder of this paper is organized as follows. In the next section, we discuss the specific market formulations we use in this study. Section 3 describes the simulation methodology we use and the results we obtain for different hypothetical market parameterizations. In Section 4 we conclude and point to future work.

2 G-commerce — Market Economies for the Grid

In formulating a computational economy for the Grid, we make two assumptions. *#1: The rel-*

ative worth of a resource is determined by its supply and the demand for it. This assumption is important because it rules out pricing schemes that are based on arbitrarily decided priorities. For example, it is not possible in an economy for an organization to simply declare what the price of its resources are and then decree that its users pay that price even if cheaper, better alternatives are available. While there are several plausible scenarios in which such Draconian policies are appropriate (e.g. users are funded to use a specific machine as part of their individual research projects), from the perspective of the Grid, the resource allocation problem under these conditions has been solved.

Further, we assume that supply and demand are functions of price, and that true relative worth is represented at the price-point where supply equals demand – that is, at market equilibrium. Conversely, at a non-equilibrium price-point (where supply does not equal demand), price either overstates or understates relative worth.

#2: Resource decisions based on self-interest are inescapable in any federated resource system. If we are to simulate a computational economy, we must ultimately hypothesize supply and demand functions for our simulated producers and consumers respectively. Individual supply and demand functions are difficult to measure at best, particularly since there are no existing Computational Grid economies which we can observe. Our admittedly less-satisfactory approach is to define supply and demand functions that represent each simulated producer and consumer's "self-interest." An individual consumer buys only if the purchase is a "good deal" for that consumer. Analogously, producers sell only when a sale is in their best interest.

In the next section, we detail the specific functions we investigate, but generally our approach relies on these two assumptions.

2.1 Producers and Consumers

To compare the efficacy of commodities markets and auctions as Grid resource allocation schemes, we define a set of simulated Grid producers and consumers representing resource providers and applications respectively. We then use the same set of producers and consumers to compare commodity and auction-based market settings.

We simulate two different kinds of producers in this study: producers of CPUs and producers of disk storage. That is, from the perspective of a resource market, there are two kinds of resources within our simulated Grids: CPUs and disks. While the results should generalize to include a variety other commodities, networks present a special problem. Our consumer model is that an application may request a specified amount of CPU and disk (the units of which we discuss below) and that these requests may be serviced by any provider regardless of location or network connectivity. Since network links cannot be combined with other resources arbitrarily, they cannot be modeled as separate commodities. We believe that network cost can be represented in terms of "shipping" costs in more complicated markets, but for the purposes of this study, we consider network connectivity to be uniform.

2.1.1 CPU Producer Model

In this study, a CPU represents a computational engine with a fixed dedicated speed. A CPU producer agrees to sell to the Grid some number of fixed "shares" of the CPU it controls. The real-world scenario for this model is for CPU owners to agree to host a fixed number of processes from the Grid in exchange for Grid currency. Each process gets a fixed, pre-determined fraction of the dedicated CPU speed, but the owner determines how many fractions or "slots" he or she is willing to sell. For example, in our study, the fraction is 10% so each CPU producer agrees to sell a fixed number (less than 10) of 10%-sized slots to the Grid. When a job occupies a CPU, it is guaranteed to get 10% of the available cycles for each slot it consumes. Each CPU, however, differs in the total number of slots it is willing to sell.

To determine supply at a given price-point, each CPU calculates

$$mean\_price = revenue/now/slots \quad (1)$$

where $revenue$ is the total amount of Grid currency (hereafter referred to as $G which is pronounced "Grid bucks"), now is an incrementing clock, and $slots$ is the total number of process slots the CPU owner is willing to support. The $mean\_price$ value is the average $G per time unit per slot the CPU has made from selling to the Grid. In our study, CPU producers will only sell if the current price of a CPU slot exceeds the $mean\_price$ value, and when they sell, they sell all unoccupied slots. That is, the CPU will sell all of its available slots when it will turn a profit (per slot) with respect to the average profit over time.

2.1.2 Disk Producer Model

The model we use for a disk producer is similar to that for the CPU producer, except that disks sell some number of fixed-sized "files" that applications may use for storage. The $mean\_price$ calculation for disk files is

$$mean\_price = revenue/now/capacity \quad (2)$$

where $capacity$ is the total number of files a disk producer is willing to sell to the Grid. If the current price for a file is greater than the $mean\_price$, a disk producer will sell all of its available files.

Note that the resolution of CPU slots and file sizes is variable. It is possible to make a CPU slot equivalent to the duration of a single clock cycle, and a disk file be a single byte. Since our markets transact business at the commodity level, however, we hypothesize that any real implementation for the Grid will need to work with larger-scale aggregations of resources for reasons of efficiency. For the simulations described in Section 3 we choose values for these aggregations that we believe reflect a market formulation that is currently implementable.

2.1.3 Consumers and Jobs

Consumers express their needs to the market in the form of jobs. Each job specifies both a size and an occupancy duration for each resource to be consumed. Each consumer also sports a budget of $G that it can use to pay for the resources needed by its jobs. Consumers are given an initial budget and a periodic allowance, but they are not allowed to hold $G over from one period until the next. This method of budget refresh is inspired by the allocation policies currently in use at the NSF Partnerships for Advanced Computational Infrastructure (PACIs). At these centers, allocations are perishable.

When a consumer wishes to purchase resources for a job, it declares the size of the request for each commodity, but not the duration. Our model is that job durations are relatively long, and that producers allow consumers occupancy without knowing for how long the occupancy will last. At the time a producer agrees to sell to a consumer, a price is fixed that will be charged to the consumer for each simulated time unit until the job completes.

For example, consider a consumer wishing to buy a CPU slot for 100 minutes and a disk file for 300 minutes to service a particular job. If the consumer wishes to buy each for a particular price, it declares to the market a demand of 1 CPU slot and 1 disk slot, but does not reveal the 100 and 300 minute durations. A CPU producer wishing to sell at the CPU price agrees to accept the job until the job completes (as does the disk producer for the disk job). Once the sales are transacted, the consumer's budget is decremented by the agreed-upon price every simulated minute, and each producer's revenue account is incremented by the same amount. If the job completes, the CPU producer will have accrued 100 times the CPU price, the disk producer will have accrued 300 times the disk price, and the consumer's budget will have been decremented by the sum of 100 times the CPU price and 300 times the disk price.

In defining this method of conducting resource transactions, we make several assumptions. First, we assume that in an actual Grid setting resource producers or suppliers will commit some fraction of their resources to the Grid, and that fraction is slowly changing. Once committed, the fraction "belongs" to the Grid so producers are not concerned with occupancy. This assumption corresponds to the behavior of some batch systems in which, once a job is allowed to occupy its processors, it is allowed to run either until completion, or until its user's allocation is exhausted. Producers are concerned, in our models, with profit and they only sell if it is profitable on the average. By including time in the supply functions, producers consider past occupancy (in terms of profit) when deciding to sell. We are also assuming that neither consumers nor producers are malicious and that both honor their commitments. In practice, this requirement assuredly will be difficult to enforce. However, if consumers and producers must agree to use secure authentication methods and system-provided libraries to gain access to Grid resources, then it will be possible.

2.1.4 Consumer Demand

The consumer demand function is more complex than the CPU and disk supply functions. Consumers must purchase enough CPU and disk resources for each job they wish to run. If they cannot satisfy the request for only one type, they do not express demand for the other. That is, the demand functions for CPU and disks are strongly correlated (although the supply functions are not). This relationship between supply and demand functions constitutes the most difficult of market conditions. Most theoretical market systems make weaker assumptions about the difference in correlation. By addressing the more difficult case, we believe our work more closely resembles what can be realized in practice.

To determine their demand at a given price, each consumer first calculates the average rate at which it would have spent $G for the jobs it has run so far if it had been charged the current price. It then computes how many $G it can spend per simulated time unit until the next budget refresh. That is, it computes

$$avg\_rate = \frac{\sum_i total\_work_i * price_i i}{now} \quad (3)$$

$$capable\_rate = \frac{remaining\_budget}{(refresh - now)} \quad (4)$$

where $total\_work_i$ is the total amount of work performed so far using commodity i, $price_i$ is the current price for commodity i, $remaining\_budget$ is the amount left to spend before the budget refresh, $refresh$ is the budget refresh time, and now is the current time. When $capable\_rate$ is greater than or equal to avg_rate, a consumer will express demand.

Unlike our supply functions, the consumer demand function does not consider past price performance directly when determining demand. Instead, consumers using this function act opportunistically based on the money they have left to spend and when they will receive more. They use past behavior only as an indication of how much work they expect to introduce and buy when they believe they can afford to sustain this rate.

Consumers, in our simulations, generate work as a function of time. We arbitrarily fix some simulated period to be a "simulated day." At the beginning of each day, every consumer generates a random number of jobs. By doing so, we hope to model the diurnal user behavior that is typical in large-scale computational settings. In addition, each consumer can generate a single new job every time step with a pre-determined probability. Consumers maintain a queue of jobs waiting for service before they are accepted by producers. When calculating demand, they compute

avg_rate and $capable\_rate$ and demand as many jobs from this queue as they can afford.

To summarize, for our G-commerce simulations:

- All entities except the market-maker act individually in their respective self-interests.
- Producers consider long-term profit and past performance when deciding to sell.
- Consumers are given periodic budget replenishments and spend opportunistically.
- Consumers introduce work loads in bulk at the beginning of each simulated day, and randomly throughout the day.

We believe that this combination of characteristics captures a reasonable set of producer and consumer traits in real Grid settings.

2.2 Commodities Markets

In a real-world commodities market, commodities are exchanged in a central location. Important features of a commodities market are that the goods of the same type brought to market by the various suppliers are regarded as interchangeable, market price is publicly agreed upon for each commodity regarded as a whole, and all buyers and sellers decide whether (and how much) to buy or sell at this price. Contrast this type of commerce with one based upon auctions, wherein each buyer and seller acts independently and contracts to buy or sell at a price agreed upon privately.

Since the goal of a computational Grid is to provide users with resources without regard to the particular supplier, it seems very natural to model a Grid economy using commodities markets. To do so, we require a pricing methodology that produces a system of price adjustments which bring about market equilibrium (i.e. equalizes supply and demand).

2.2.1 Pricing in Commodities Markets: Results of Economic Research

Our model is an example of an*exchange economy*, namely a system involving *agents* (producers and consumers), and several commodities. Each agent is assumed to control a sufficiently small segment of the market. In other words, the individual behavior of any one agent will not affect the system as a whole appreciably. In particular, prices will be regarded as beyond the control of the agents. Given a system of prices, then, each agent decides upon a course of action, which may consist of the sale of some commodities and the purchase of others with the proceeds. Thus we define supply and demand functions for each commodity, which are functions of the aggregate behavior of all the agents. These are determined by the set of market prices for the various commodities.

Naturally, we use the language of vectors for price, supply, and demand; each of these will be an n-vector, where n is the number of commodities, of non-negative real numbers. Observe that given a *commodity bundle*, that is an $n - vector$ of quantities $\mathbf{x} = x_1, ..., x_n$ of the commodities, and a price vector $\mathbf{p}$ the value of the bundle is equal to $\mathbf{p} \cdot \mathbf{x}$. For given price vector $\mathbf{p}$, define the *excess demand* $\mathbf{z} = \mathbf{z}(\mathbf{p})$ to be the difference of the demand and supply vectors for this price level. *Equilibrium* for the economy is established when supply is equal to demand; in other words, a price vector $\mathbf{p}$ is an equilibrium price when $\mathbf{z}(\mathbf{p}) = \mathbf{0}$. It should be noted that, for our purposes, currency will be regarded as another commodity. Thus a producer of a non-currency commodity (CPU or disk for the purposes of this paper) will simply be regarded as a "consumer" of currency; presumably, the currency will be used in some way for the benefit of the producer.

In general equilibrium theory, there are three hypotheses made on the function $\mathbf{z}$: *homogeneity*, *continuity*, and adherence to *Walras' Law*. Homogeneity means that only the ratios between prices are important to how commodities are exchanged.

That is, $\mathbf{z}(\lambda\mathbf{p}) = \mathbf{z}(\mathbf{p})$ for any positive number λ. This relationship is naturally true, since currency is regarded as a commodity. Continuity is the property that excess demand is a continuous function of the prices, which cannot hold literally in our situation, due to the indivisibility of the commodities. However, we assume that the number of agents is large enough that all functions may be approximated by continuous functions of continuous variables. Finally, Walras' Law states that for any price, $\mathbf{z}(\mathbf{p}) \cdot \mathbf{p} = 0$. This assumption is justified as follows: When each agent is supplying the same total value as that agent is demanding, the value of the total supply bundle $\mathbf{s}$ is equal to that of the total demand bundle $\mathbf{d}$. Thus, as observed above, $\mathbf{p} \cdot \mathbf{s} = \mathbf{p} \cdot \mathbf{d}$, and therefore $\mathbf{p} \cdot \mathbf{z} = \mathbf{p} \cdot (\mathbf{d} - \mathbf{s}) = \mathbf{0}$. Walras' Law will apply as long as demand is *locally non-satiated*, that is, given a level of consumption, there is always a preference for greater consumption (price not being an object).

When these assumptions have been met, an equilibrium price vector has been proven to exist via topological methods, namely the *Brouwer fixed-point theorem* (see [13], Chapter 5, for the result in its original form, or a remarkably clear exposition in [15], Chapter 6). These methods are non-constructive, so that the problem remains to find a method of price adjustment that brings about equilibrium or at least *approximates equilibrium within reasonable tolerances*.

A few words on this last point are in order. From a purely "engineering" standpoint, reaching precise economic equilibrium is surely impossible. Thus we must content ourselves with the more modest goal of producing a price vector for which the excess demands are all close to 0. Since the excess demand functions can be quite general, it is always possible that there exists a price vector which produces excess demands which are all within a prescribed tolerance of 0 and yet is not close to an actual equilibrium point; further, there is no "engineering" method which will distinguish this from a point which really is very near to an equilibrium price. Even Scarf's algorithm, described below, which has erroneously been called a "constructive version of the Brouwer fixed-point theorem," is only guaranteed to produce points which are approximate equilibria in the first sense. Thus we will use the phrase "approximate equilibrium" to refer to a price which makes the excess demands all close to 0 without judging whether it lives near a genuine equilibrium point. In any event, the theoretical existence of an equilibrium price guarantees the existence of approximate equilibria. Moreover, approximate equilibria are valuable: If the market is approximately cleared, then the economy is doing a good job of distributing goods.

Walras in [37] suggested a process called *tâtonnement* ("groping") by which real-world markets come to equilibrium. With *tâtonnement*, each individual price is raised or lowered according to whether that commodity's excess demand is positive or negative. Then, new excess demands are measured, and the process is iterated. While it was suggested only as a "behavioral" explanation as to how real-world markets reach equilibrium, *tâtonnement* formed the basis for early attempts to prove the existence of equilibrium. It is now known that *tâtonnement* does not in general lead to a convergent process; Scarf in [30] produced a very simple example for which there is a unique equilibrium but for which, from almost every starting point, the *tâtonnement* process oscillates for all time. In fact, *tâtonnement* does bring about convergence to an equilibrium price vector under the very strong hypothesis of *gross substitutes*, which states that increasing the j^{th} price while holding the others constant will bring about an increase in excess demand in all commodities other than the j^{th}. Unfortunately, for typical Grid applications, the hypothesis of gross substitutes does not hold, because different commodities are often complementary. (For example, an application may need both CPU and disk in order to execute. If the price for CPUs is too high, then the application's demand for disks will be lower instead of higher.)

There are several different approaches to the problem of finding an algorithm for adjusting prices which will lead to equilibrium. *Scarf's algorithm* (see [31]) works roughly as follows: Suppose that there are $n + 1$ commodities, and normalize the prices so that their sum is always equal to 1. The set of possible price vectors thus forms an n-dimensional *simplex* in $\mathbf{R}^{n+1}$ (the *price simplex*). Scarf then divides this simplex into a large number of subsimplices and shows that there exists a subsimplex any of whose points provides an approximate equilibrium price. He also provides an explicit formula for how fine to make the subdivision in order to produce an excess demand within a pre-specified tolerance. Merrill [23] gives an important improvement to Scarf's algorithm which makes it far more attractive from a computational standpoint. A different sort of refinement of this idea is to be found in Eaves' algorithm with "continuous refinement of grid size" [14].

A second approach, advocated by Smale in [32], is more in the spirit of multivariable calculus and is more dynamic in the sense that it aims to produce a trajectory for the prices to follow. In Smale's method, the prices are normalized by fixing one of the commodities (the *numeraire*) to have price 1; in our case, this commodity will be the currency. Further, suppose that there are n other commodities, so that the set of possible prices forms the positive orthant in $\mathbf{R}^n$. Form the $n \times n$ matrix

$$D_{\mathbf{z}}(\mathbf{p}) = \left(\frac{\partial \mathbf{z_i}}{\partial \mathbf{p_j}}\right).$$

Now define the *global Newton* ordinary differential equation

$$D_{\mathbf{z}}(\mathbf{p})\frac{\mathbf{dp}}{\mathbf{dt}} = -\lambda \mathbf{z}(\mathbf{p}) \qquad (5)$$

where λ is a constant which has sign equal to $(-1)^n$ times the sign of the determinant of $D_{\mathbf{z}}(\mathbf{p})$. (For contrast, note that the *tâtonnement* process is encapsulated in the differential equation $\frac{d\mathbf{p}}{dt} = \mathbf{z}$. Thus the global Newton may be regarded as a more sophisticated version of *tâtonnement* which takes into account the interdependencies of the way demands for the various commodities interact with the various prices.) Smale proves that, under boundary conditions which are justifiable on the basis of the desirability of the commodities, almost every maximal solution of the global Newton equation starting sufficiently near to the boundary of the positive orthant of $\mathbf{R}^n$ (or to ∞) will converge to the set of equilibrium prices.

Note that except under strong hypotheses, most commonly gross substitutes, the theory does not guarantee that there is a *unique* equilibrium price vector. However, there is a useful result along these lines as follows: Define a *regular* equilibrium to be one for which the matrix $D_{\mathbf{z}}(\mathbf{p})$ defined above is nonsingular. Then according to [22], Theorem 5.4.2, a regular equilibrium price is *locally* unique in the sense that it is the only one in some open subset of the space of price vectors.

2.2.2 Price Adjustment Schemes

Herein we examine the results of using several price adjustment schemes in simulated computational market economies. Smale's method is not possible to use directly for a number of reasons. First, any actual economy is inherently discrete, so the partial derivatives in equation 5 do not exist, strictly speaking. Second, given the behavior of the producers and consumers described above, there are threshold prices for each agent that bring about sudden radical changes in behavior, so that a reasonable model for excess demand functions would involve sizeable jump discontinuities. Finally, the assumptions in Smale's model are that supply and demand are functions of price only and independent of time, whereas in practice there are a number of ways for supply and demand to change over time for a given price vector.

Observe that taking $\lambda = 1$ and applying the Euler discretization at positive integer values of

t reduces this process to the Newton-Raphson method for solving $\mathbf{z}(\mathbf{p}) = \mathbf{0}$; this observation explains the term "global Newton."

Implementing Smale's method: As observed above, obtaining the partial derivatives necessary to carry out Smale's process in an actual economy is impossible; however, within the framework of our simulated economy, we are able to get good approximations for the partials at a given price vector by polling the producers and consumers. Starting with a price vector, we find their preferences at price vectors obtained by fixing all but one price and varying the remaining price slightly, thus achieving a "secant-line" approximation for each commodity separately; we then substitute these approximations for the values of the partial derivatives in the matrix $D_{\mathbf{z}}(\mathbf{p})$, discretize with respect to time, solve Equation 5 for the increment $d\mathbf{p}$ to get our new price vector, and iterate. We will refer, conveniently but somewhat inaccurately, to this price adjustment scheme as *Smale's method*.

The First Bank of G: The drawback to the above scheme is that it relies on polling the entire market for aggregate supply and demand repeatedly to obtain the partial derivatives of the excess demand functions. If we were to try and implement Smale's method directly, each individual producer and consumer would have to be able to respond to the question "how much of commodity x would you buy (sell) at price vector $\mathbf{p}$?" In practice, producers and consumers may not be able to make such a determination accurately for all possible values of $\mathbf{p}$. Furthermore, even if explicit supply and demand functions are made into an obligation that all agents must meet in order to participate in an actual Grid economy, the methodology clearly will not scale. For these reasons, in practice, we do not wish to assume that such polling information will be available.

A theoretically attractive way to circumvent this difficulty is to approximate each excess demand function z_i by a polynomial in $p_1, p_2, ..., p_n$ which fits recent price and excess demand vectors and to use the partial derivatives of these polynomials in Equation 5. In simulations, this method does not, in general, produce prices which approach equilibrium. The *First Bank of G* is a price adjustment scheme which both is practicable and gives good results; this scheme involves using *tâtonnement* (see above) until prices get "close" to equilibrium, in the sense that excess demands have sufficiently small absolute value, and then using the polynomial method for "fine tuning." Thus, the First Bank of G approximates Smale's method but is implementable in real-world Grid settings since it hypothesizes excess demand functions and need not poll the market for them. Our experience is that fairly high-degree polynomials are required to capture excess demand behavior with the sharp discontinuities described above. For all simulations described in Section 3, we use a degree 17 polynomial.

2.3 Auctions

Auctions have been extensively studied as resource allocation strategies for distributed computing systems. In a typical auction system (e.g. [11, 35, 25, 6]), resource producers (typically CPU producers) auction themselves using a centralized auctioneer and sealed-bid, second-price auctions. That is, consumers place one bid with the auctioneer, and in each auction, the consumer with the highest bid receives the resource at the price of the second-highest bidder. This is equivalent to "just" outbidding the second-highest bidder in an open, multi-round auction, and encourages consumers to bid what the resource is worth to them (see [6] for further description of auction variants).

When consumers simply desire one commodity, for example CPUs in Popcorn [25], auctions provide a convenient, straightforward mechanism for clearing the marketplace. However, the assumptions of a Grid Computing infrastructure pose a few difficulties to this model. First, when an application (the consumer in a Grid Computing scenario) desires multiple commodities, it must place simultaneous bids in multiple auctions, and may only be successful in a few of these. To do so, it must expend currency on the resources that it has obtained while it waits to obtain the others. This expenditure is wasteful, and the uncertain nature of auctions may lead to inefficiency for both producers and consumers.

Second, while a commodities market presents an application with a resource's worth in terms of its price, thus allowing the application to make meaningful scheduling decisions, an auction is more unreliable in terms of both pricing and the ability to obtain a resource, and may therefore result in poor scheduling decisions and more inefficiency for consumers.

To gain a better understanding of how auctions fare in comparison to commodities markets, we implement the following simulation of an auction-based resource allocation mechanism for computational grids. At each time step, CPU and disk producers submit their unused CPU and file slots to a CPU and a disk auctioneer. These are accompanied by a minimum selling price, which is the average profit per slot, as detailed in Section 2.1.1 above. Consumers use the demand function as described in Section 2.1.3 to define their bid prices, and as long as they have money to bid on a job, and a job for which to bid, they bid on each commodity needed by their oldest uncommenced job.

Once the auctioneers have received all bids for a time step, they cycle through all the commodities in a random order, performing one auction per commodity. In each auction, the highest-bidding consumer gets the commodity if the bid price is greater than the commodity's minimum price. If there is a second-highest bidder whose price is greater than the commodity's minimum price, then the price for the transaction is the second-highest bidder's price. If there is no such second-highest bidder, then the price of the commodity is the average of the commodity's minimum selling price and the consumer's bid price. When a consumer and commodity have been matched, the commodity is removed from the auctioneer's list of commodities, as is the consumer's bid. At that point, the consumer can submit another bid to that or any other auction, if desired. This situation occurs when a consumer has obtained all commodities for its oldest uncommenced job, and has another job to run. Auctions are transacted in this manner for every commodity, and the entire auction process is repeated at every time step.

Note that this structuring of the auctions means that each consumer may have at most one job for which it is currently bidding. When it obtains all the resources for that job, it immediately starts bidding on its next job. When a time step expires and all auctions for that time step have been completed, there may be several consumers whose jobs have some resources allocated and some unallocated, as a result of failed bidding. These consumers have to pay for their allocated resources while they wait to start bidding in the next time step.

While the auctions determine transaction prices based on individual bids, the supply and demand functions used by the producers and consumers to set ask and bid price are the same functions we use in the commodities market formulations. Thus, we can compare the market behavior and individual producer and consumer behavior in both auction and commodity market settings.

3 Simulations and Results

We compare commodities markets and auctions using the producers and consumers described in Section 2.1 using two overall market settings. In the first, which we term *under-demand*, producers are capable of supplying enough resource to service all of the jobs consumers can afford. Recall that our markets do not include resale components. Consumers do not make money. Instead, $G are given to them pe-

CPUs	100
disks	100
CPU slots per CPU	[2 .. 10]
disk files per disk	[1 .. 15]
CPU job length	[1 .. 60] time units
disk job length	[1 .. 60] time units
simulated day	1440 time units
allowance period	[1 .. 10] days
jobs submitted at day-break	[1 .. 100]
new job probability	10%
allowance	10^6 $G
Bank of G Polynomial Degree	17
λ factor	.01

Table 1. Invariant simulation parameters for this study

riodically much the in the same way that PACIs dole out machine-time allocations. Similarly, producers do not spend money. Once gathered, it is hoarded or, for the purposes of the economy, "consumed." The under-demand case corresponds to a Grid economy in which the allocations exceed what is necessary (in terms of user demand) to allocate all available resources. Such a situation occurs when the rate that $G are allocated to consumers is greater than the rate at which they introduce work to the Grid. In the *over-demand* case, consumers wish to buy more resource than is available. New jobs are generated fast enough to keep all producers almost completely busy, thereby creating a work back-log.

Table 1 completely describes the invariant simulation parameters we use for both under- and over-demand cases. For all ranges (e.g. slots per CPU), uniform pseudo-random numbers are drawn from between the given extrema. For the under-demand simulation, we define 100 consumers to use the 100 CPUs and disks. Each consumer submits a random number of jobs (between 1 and 100) at every day-break, and has a 10% chance of submitting a new job every time unit. The over-demand simulation specifies 500 of the same consumers, with all other parameters held constant.

Using our simulated markets, we wish to investigate three questions with respect to commodities markets and auctions.

1. *Do the theoretical results from Smale's work [33] apply to plausible Grid simulations?*

2. *Can we approximate Smale's method with one that is practically implementable?*

3. *Are auctions or commodities markets a better choice for Grid computational economies?*

Question (1) is important because if Smale's results apply, they dictate that an equilibrium price-point must exist (in a commodity market formulation), and they provide a methodology for finding those prices that make up the price-point. Assuming the answer to question (1) is affirmative, we also wish to explore methodologies that achieve or approximate Smale's results, but which are implementable in real Grid settings. Lastly, recent work in Grid economies [1, 18, 28] and much previous work in computational economic settings [12, 26, 5, 36] has centered on auctions as the appropriate market formulation. We wish to investigate question (3) to determine whether commodities markets are a viable alternative and how they compare to auctions as a market-making strategy.

3.1 Market Conditions, under-demand case

Figure 1 shows the CPU and disk prices for Smale's method in our simulated Grid economy over $10,000$ time units. The diurnal nature of consumer job submission is evident from the price fluctuations. Every 1440 "minutes" each consumer generates between 1 and 100 new jobs causing demand and prices to spike. However, Smale's method is able to find an equilibrium

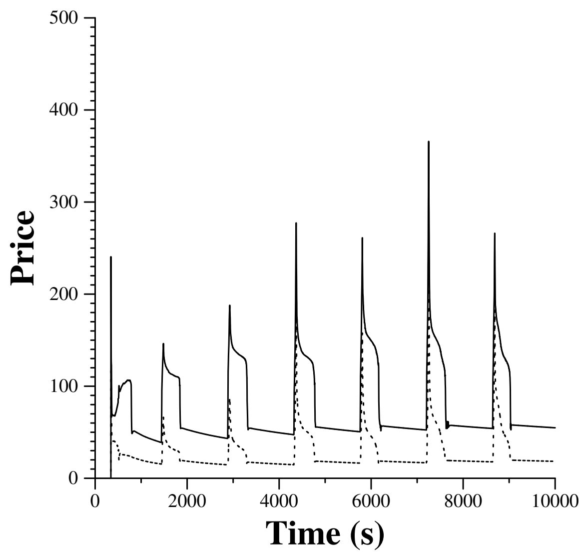

Figure 1. Smale's prices for the under-demand case. Solid line is CPU price, and dotted line is disk price in $G

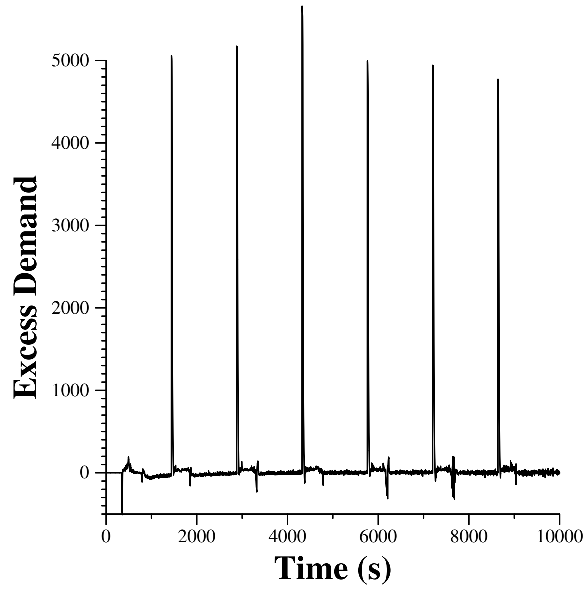

Figure 2. Smale's CPU excess demand for the under-demand case. The units are CPU slots.

price for both commodities quickly, as is evidenced in Figure 2. Notice that the excess demand spikes in conjunction with the diurnal load, but is quickly brought near zero by the pricing shown in Figure 1 where it hovers until the next cycle. Figure 3 shows excess demand for disk during the simulation period. Again, approximate market equilibrium is quickly achieved despite the cyclic and non-smooth aggregate supply and demand functions implemented by the producers and consumers.

In Figure 4 we show the pricing determined by our engineering approximation to Smale's method — the First Bank of G. The First Bank of G pricing closely approximates the theoretically achievable results generated by Smale's method in our simulated environment. The Bank, though, does not require polling to determine the partial derivatives for the aggregate supply and demand functions. Instead, it uses an iterative polynomial approximation that it derives from simple observations of purchasing and consumption. Thus it is possible to implement the First Bank of G for use in a real Grid setting without polling Grid producers or consumers for their supply and demand functions explicitly. Figures 5 and 6 show excess demand measures generated by First Bank of G pricing over the simulated period. While the excess demands for both commodities are not as tightly controlled as with Smale's method, the First Bank of G keeps prices very near equilibrium.

The pricing determined by auctions is quite different, however, as depicted in Figures 7 and 8 (we show CPU and disk price separately as they are almost identical and obscure the graph when overlayed). In the figure, we show the average price paid by all consumers for CPU during each auction round. We use the average price for all auctions as being representative of the "global" market price. Even though this price is smoothed as an average (some consumers pay more and some pay less during each time step), it shows considerably more variance than prices set by the commodities market. The spikes in workload are not reflected in the price, and the variance seems to increase (i.e. the price becomes less stable) over time.

Excess demand for an auction is more difficult

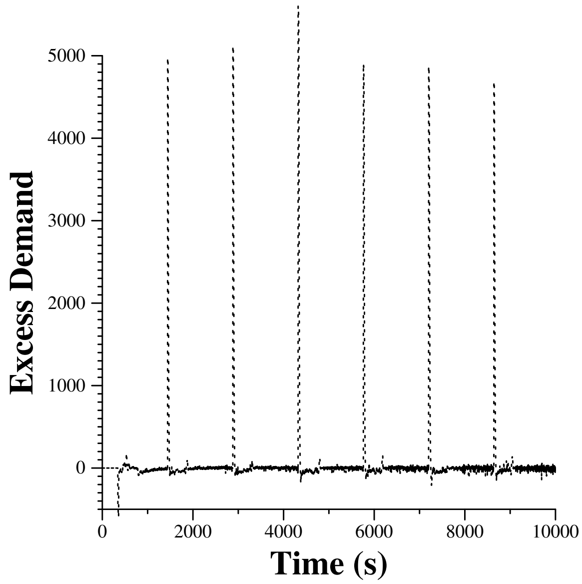

Figure 3. Smale's disk excess demand for the under-demand case. The units are simulated file units.

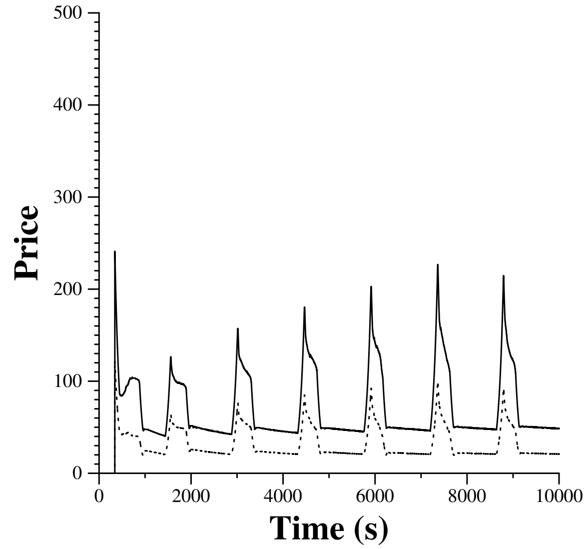

Figure 4. First Bank of G prices for the under-demand case. Solid line is CPU price, and dotted line is disk price in $G

to measure since prices are negotiated between individual buyers and sellers. As an approximation, we consider the sum of unsatisfied bids and the number of auctions that did not make a sale as a measure of market disequilibrium. Under this assumption, the market is in equilibrium when all bids are satisfied (demand is satisfied) and all auctioned goods are sold (supply is exhausted). Any surplus goods or unsatisfied bids are "excess." While is does not make sense to assign a sign to these surpluses (surplus supply, for example, may not be undemanded supply) in the way that we can with aggregate supply and demand in a commodity market, in absolute value this measure captures distance from equilibrium. Hence we term it *absolute excess demand*.

In Figure 9 we show this measure of excess demand for CPUs in the under-demanded auction. Figure 10 shows the same data as in Figure 5 from the First Bank of G, but in absolute value. While the First Bank of G shows more variance in absolute excess demand, it achieves approximate equilibrium and sustains it over relatively long periods. By contrast, the auction sets prices that never satisfy the market. Strangely, the auction comes closest to equilibrium when demand spikes at each day-break. We are working to understand this behavior and will report on it as part of our future work.

From these simulation data we conclude that Smale's method is appropriate for modeling a hypothetical Grid market and that the First Bank of G is a reasonable (and implementable) approximation of this method. These results are somewhat surprising given the discrete and sharply changing supply and demand functions used by our producers and consumers. Smale's proofs assume continuous functions and readily available partial derivatives. We also note that auctioneering, while attractive from an implementation standpoint, does not produce stable pricing or market equilibrium. If Grid resource allocation decisions are based on auctions, they will share this instability and lack of fairness. A commodities market formulation, at least in simulation, performs better *from the standpoint of the Grid as a whole*. These results agree with those reported in [36] which indicate that auctions are locally

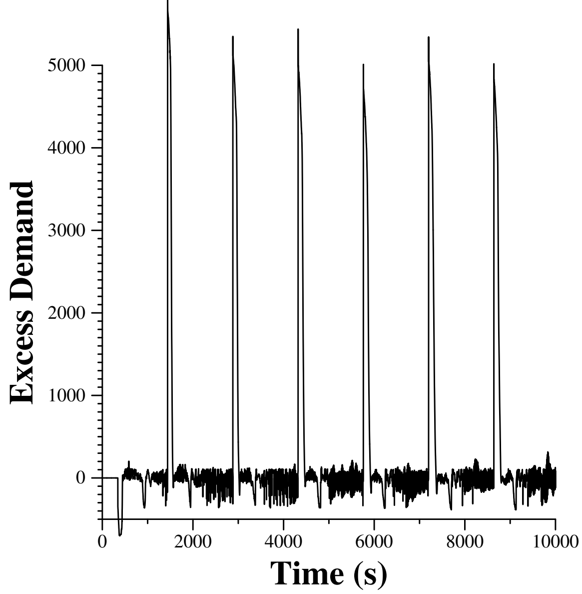

Figure 5. First Bank of G CPU excess demand for the under-demand case. The units are CPU slots.

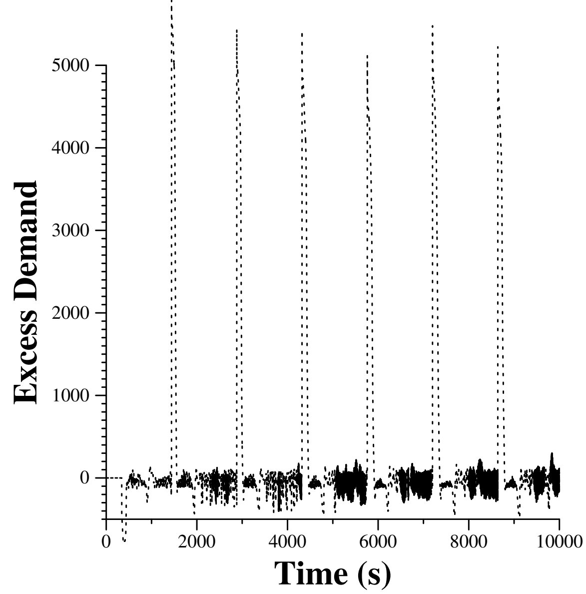

Figure 6. First Bank of G disk excess demand for the under-demand case. The units are simulated file units.

advantageous, but may exhibit volatile emergent behavior system wide.

3.2 Market Conditions, over-demand case

For the over-demand market case, we increase the number of consumers to 500 leaving all other parameters fixed. As in the under-demand case, Smale's method produces a stable price series which the Bank of G is able to approximate but which auctions are unable to match. We omit the bulk of the results in favor of examining the behavior of both Smale's method and the Bank of G as they converge to an approximate economic equilibrium.

Figure 11 shows the pricing information using Smale's method for the over-demand market, and Figure 12 shows the prices determined by the First Bank of G. Note that Smale's method determines a higher price for disk than CPU and that the First Bank of G chooses a significantly higher price for CPU, but a lower price for disk. Intuitively one expects a higher price for CPU than disk since CPU is the "rarer" commodity in our simulation. The Bank of G would seem to correctly identify CPU as the scarcer commodity by setting a higher price for it. Nonetheless, excess demand graphs (Figures 13 and 14) for CPU indicate that both solution methods are centered on market equilibrium. While it is difficult to read from the graphs (we use a uniform scale so that all graphs of a certain type in this study may be compared), the mean excess demand for the data shown in Figure 13 is 52.4, and the the First Bank of G data in Figure 14, the mean excess demand is 25.6. Both of these values are near enough to zero to constitute approximate equilibria for our purposes.

3.3 Multiple Equilibria

We wish to examine more closely the phenomenon of apparent multiple economic equilibria within our simulated market. In particular, we claim that both the solutions arrived at by Smale's method and by the Bank of G are valid approximations of economic equilibria and may in fact be approximations of actual equilibria. To facili-

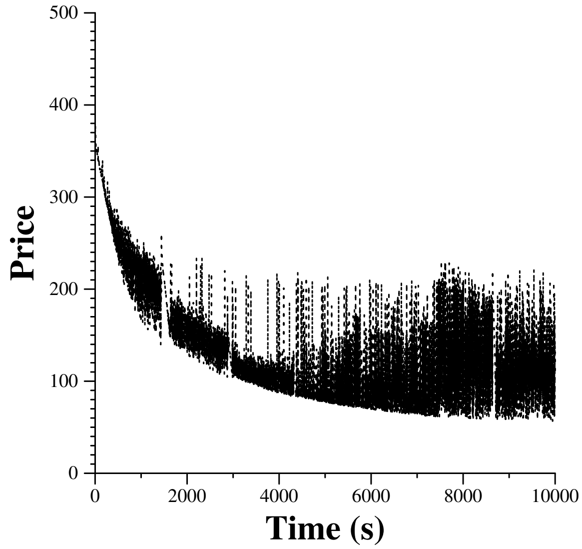

Figure 7. Auction prices for the under-demand case, average CPU price only, in $G

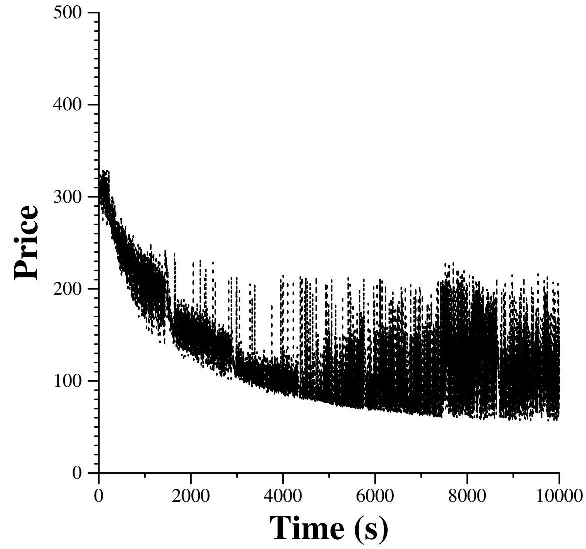

Figure 8. Auction prices for the under-demand case, average disk price only, in $G

tate our examination, we will examine the aggregate supply and demand functions over all producers and consumers at particular points in the simulation. To do so, we freeze the simulation after it has reached approximate equilibrium and then query the producers and consumers for supply and demand values over a range of prices. This technique produces a profile of the macroeconomic supply and demand curves which should reveal equilibria at their intersection points.

Recall that, in our simulated economy, CPU and disk are highly complementary. Since demand for one commodity is not independent of demand for the other, we must generate families of aggregate demand curves, in which the price of one commodity is held constant while the price of the other commodity is varied over the specified range. Each generated demand curve in a family is associated with a single fixed price for the other commodity. Then, the fixed price is incremented and another aggregate supply curve is generated. This process continues until the fixed price also reaches the upper limit of the specified price range. If generating aggregate demand curves for the CPU commodity, for example, the simulator produces one curve per price of the disk commodity.

Note that, together, these families of curves form a three-dimensional surface for each commodity in which the axes are CPU price, disk price, and demand. That is, for each ordered pair of CPU and disk prices there is a corresponding CPU demand value. Similarly, a second surface is formed from the CPU price, disk price, and disk demand coordinates.

In contrast, the supply of a commodity in our economy is never correlated with the supply of another commodity and varies only with price, so it is not necessary to produce families of aggregate supply curves. Instead, we produce a single supply curve by freezing the simulation and varying the price of a commodity over some range while querying for aggregate supply at each new price value.

Figures 15, 16, 17 and 18 show aggregate supply and demand curves for CPU and disk in the over-demand case. Both Smale's method and the Bank of G are shown. The simulation freezes at time slice 2000 and produces aggregate curves. Rather than representing the three-

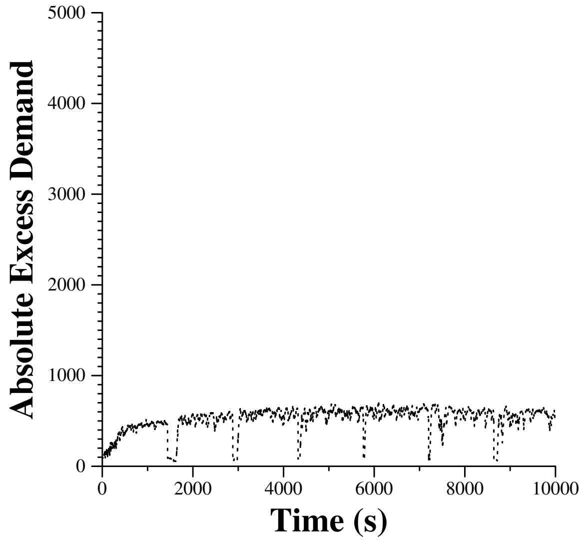

Figure 9. Auction absolute excess demand for CPU in the under-demand case. The units are CPU slots.

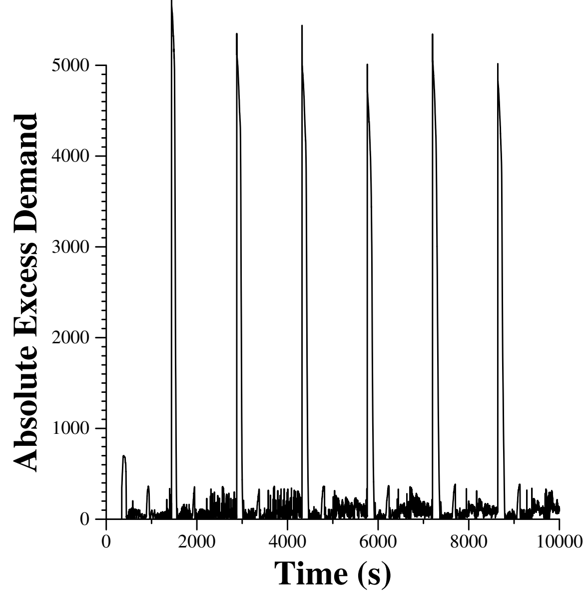

Figure 10. First Bank of G absolute excess demand for CPU in the under-demand case. The units are CPU slots.

dimensional surface of prices and demand (which is difficult to represent without the use of color), we depict the relationships in terms of a labeled two-dimensional projection.

In Figure 15, the x axis represents CPU price and the y axis corresponds to CPU units (either of supply or demand). Each nearly vertical curve is a CPU demand function relating CPU price to CPU demand for a given disk price (shown as a label on each curve at the top of the graph). We only show CPU demand curves at 10 $G increments, although one exists for each possible price. As a thick gray line, we show the CPU demand curve that corresponds to the disk price ($G 211.4 in the figure) that Smale's method determined at the time we froze the simulation. The thick dotted line near the bottom of the graph shows the CPU supply curve as a function of price. The x coordinate of the price point where the CPU demand curve (shown in thick gray) intersects the CPU supply curve (dotted black) corresponds to the approximate equilibrium price for CPU within simulated economy at the given time step. The solid circle on the graph shows the price-point that Smale's method determined for the same time step. If the circle covers the intersection (as it does in Figure 15) the price adjustment strategy has correctly determined an approximate equilibrium price for the economy.

Similarly, in Figures 16, 17, and 18 the demand curves are labeled with the fixed price of the other commodity used to produce the curve: for example, one CPU demand curve shown corresponds to holding the price of disk to $G 200 while varying the price of CPU. Since demand for one type of commodity is tied to demand for the other, the demand curve families for both disk and CPU tend to be similar. Only a few demand curves in the family are shown, but it is important to note that an infinity of such curves exist, forming a demand curve surface. Also shown in Figures 16, 17 and 18 are the aggregate supply curves for each commodity, shown in a thick dotted line. Supply of both commodities remains constant across the price range shown, because all simulated suppliers are "producing" at maximum capacity. No matter how high the price may be set, no more CPU or disk is available within the

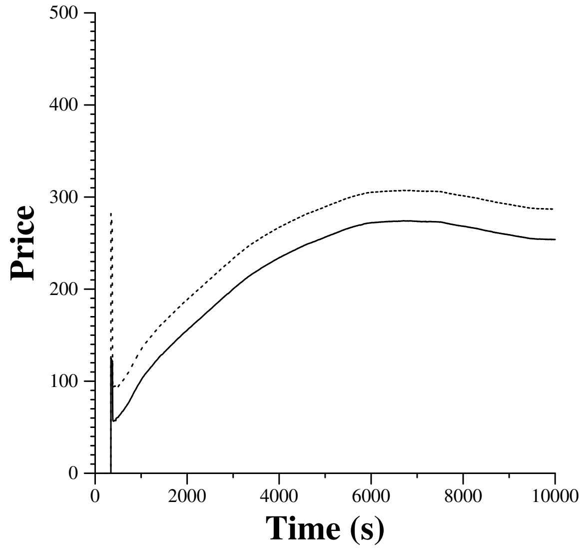

Figure 11. Smale's CPU and disk prices for the over-demand case. Solid line is CPU price, dotted line is disk price, and the units are $G.

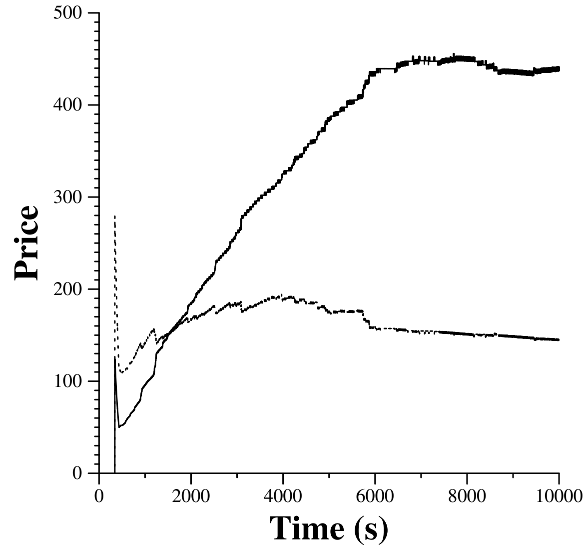

Figure 12. First Bank of G CPU and disk prices for the over-demand case. Solid line is CPU price, dotted line is disk price, and the units are $G.

economy.

Figures 15 and 16 have been obtained by running Smale's method until it reaches an approximate equilibrium at a CPU price of about $G 161.8 and a disk price of about $G 211.4, which are marked as heavy dots on the respective graphs. For Figure 15, the disk prices were then artificially fixed at various values and the CPU demand curves, labelled by disk price across the top of the graph, were generated by polling the consumers. Again, in principle there exist demand curves for all possible disk prices; we have shown only multiples of $G 10. For Figure 16, the roles of the commodities are reversed. Note that supply of each commodity is a function of that commodity's price alone, so that only one supply curve exists on each of the graphs.

Figure 15 shows that the CPU market is cleared for a CPU price of about $G 161 (read from the horizontal axis) and a disk price of about $G 211 (read from the family of curves). Similarly, one finds from the heavy dot in Figure 16 that the disk market is cleared for about the same respective prices for disk and CPU. However, from the graphs it is possible to find other price combinations which clear each market separately. For example, it is evident from Figure 15 that a CPU price of about $G 175 and a disk price of $G 200 will also clear the market, since the CPU demand curve corresponding to a disk price of $G 200 intersects the supply curve at a point where the CPU price is about $G 175. Now look at Figure 16. It seems that a disk price of about $G 200 and a CPU price of $G 175 will clear the disk market as well! Moreover, within the range of prices shown on the two graphs, it looks as though *any* price vector which clears one market also clears the other market as well, or at least very nearly so. Thus it would appear that there is a whole connected curve of market equilibria for our economy.

From a "behavioral" standpoint, this set of relationships between supply, demand, and price may be explained as follows: The two commodities are extremely complementary, meaning that they are used together rather than in competition with one another. As long as the consumers have some

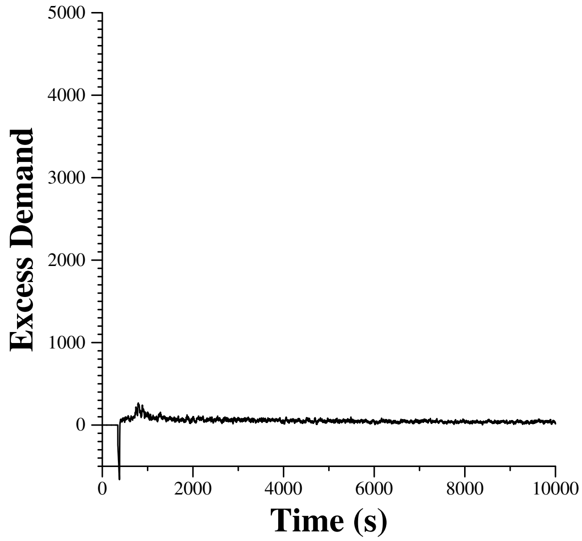

Figure 13. Smale's CPU excess demand for the over-demand case. The units are CPU slots.

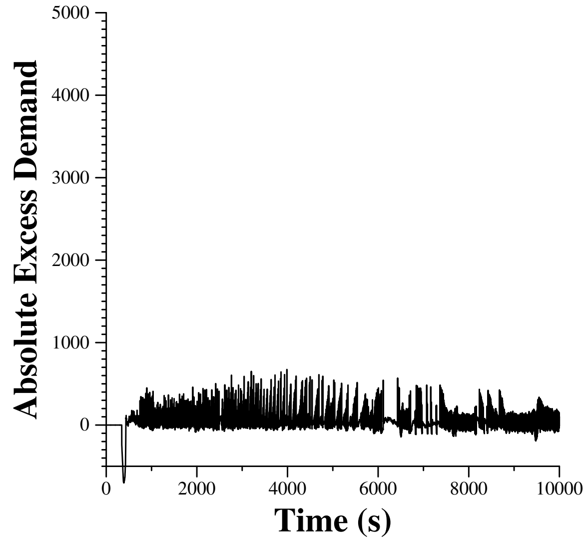

Figure 14. First Bank of G CPU excess demand for the over-demand case. The units are CPU slots.

choice as to which jobs to perform (as they do in the overdemand case, since job queues never clear), and as long as the price of one commodity is lowered in conjunction with a rise in the price of the other, it is always possible for the consumers to make purchasing decisions which allow them to spend their allotment, choosing, if the prices are different, to complete jobs which are more intensive in the commodity which is less expensive.

It is interesting to note that in this case one can find the point in the theory where the hypotheses which rule out non-locally-unique equilibria break down. It is apparent that in our experiments the two commodities are so complementary that the demand functions shift in the same way in response to increases in either price. Thus the columns of the Jacobian matrix $D_{\mathbf{z}}(\mathbf{p})$ of partial derivatives of the excess demand with respect to price are (approximately) linearly dependent at equilibrium. By definition, then, the equilibrium is not *regular*, and therefore it need not be locally unique according to the theory (*Cf.* Section 2.2.1).

In any event, it would seem that these apparent multiple equilibria arise not because of any anomalies in our method per se, but rather because our experimental economy is so very simple as to consist of only two commodities (plus currency) which are essentially in perfect complementarity. One would expect that, as the model becomes more complex, this particular sort of difficulty will vanish. Further, even in the presence of multiple equilibria, each of our price adjustment schemes continued to behave in such a way as to produce long-term stability and approximate market-clearing. This is all that one can practically hope for, since even in well-behaved ("regular") economies, there may be multiple (isolated) equilibria with no rational basis for choice among them.

Our implementation of Smale's technique, then, finds a valid equilibrium price from among a space of possible equilibria. The Bank of G also finds a valid price solution, albeit a different one from Smale's technique. In Figures 17 and 18, we show the supply and demand curve families as well as their price solutions for the Bank of G. Note again that the prices correspond to a

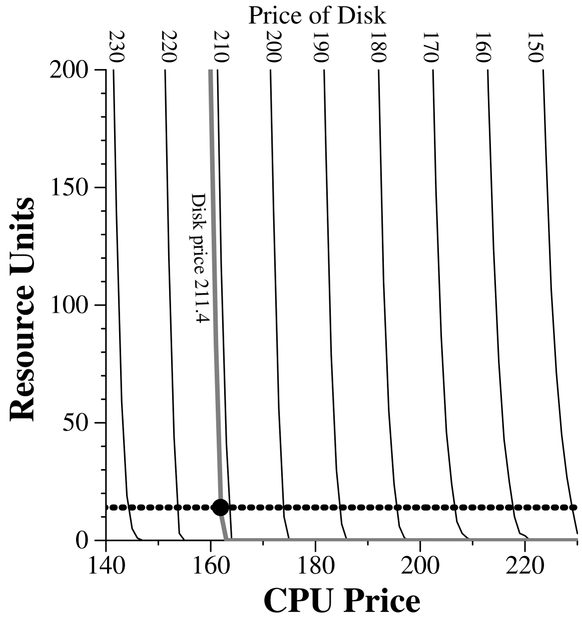

Figure 15. CPU aggregate supply and demand curves for Smale's method, over-demand case, iteration 2000.

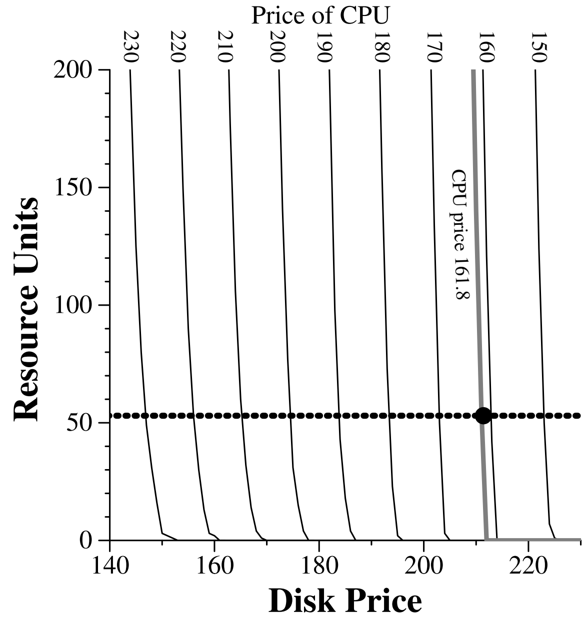

Figure 16. Disk aggregate supply and demand curves for Smale's method, over-demand case, iteration 2000.

global equilibrium; the CPU price point lies at the intersection of the CPU supply curve and the CPU demand curve corresponding to disk price of $G 166. Since the market is in an over-demand situation, resource consumers have no choice in the mix of jobs they run. Rather, they can run only jobs for which some supply is available. Consumers' jobs queue waiting to be serviced, and this queue contains a mixture of CPU- and disk-intensive jobs. Thus, from the standpoint of global equilibrium, additional disk supply and additional CPU supply are interchangeable; there is ample demand to utilize either. The market is free to choose any balance between CPU and disk price so long as the aggregate supply of either commodity remains fully utilized.

From this basis the price inversion of CPU and disk between the Smale and Bank of G over-demand simulations is easy to understand. Both methods clear the market and control excess demand. Valid price solutions are necessary to accomplish such control, and both techniques find such solutions. It is intuitively uncomfortable for Smale's technique to arrive at higher prices for more plentiful commodities, but such behavior is sound from an economic standpoint.

Note that in every case (Figures 15, 16, 17, and 18) the respective method (either Smale or Bank of G) determines a price that is at or very close an approximate equilibrium price for the economy.

As noted above, the price vector solution space for two commodities can effectively be viewed as a 3 dimensional plot of total absolute excess demand versus the price of both commodities. Total absolute excess demand is in this case defined as the sum of the absolute value of the excess demand for both commodities, and can be used as a measure of closeness to economic equilibrium. In Figures 19 and 20 we show this space of price solutions for the over-demand case. For clarity, only the point of minimum excess demand for each demand curve is shown. These points form a line in price/excess demand space along which approxi-

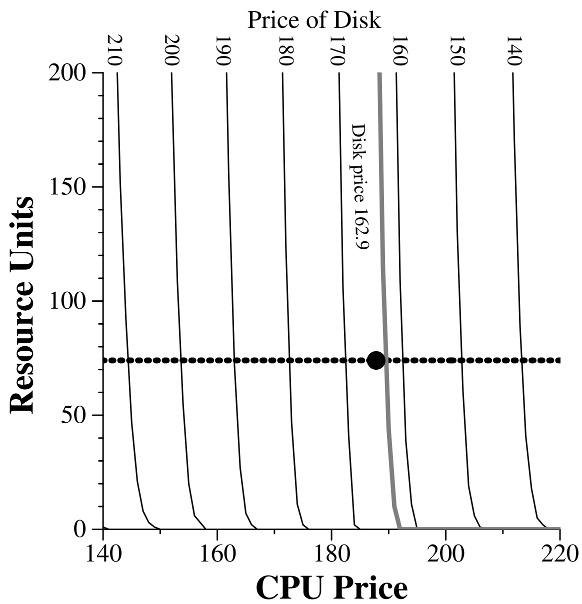

Figure 17. CPU aggregate supply and demand curves for the Bank of G, over-demand case, iteration 2000.

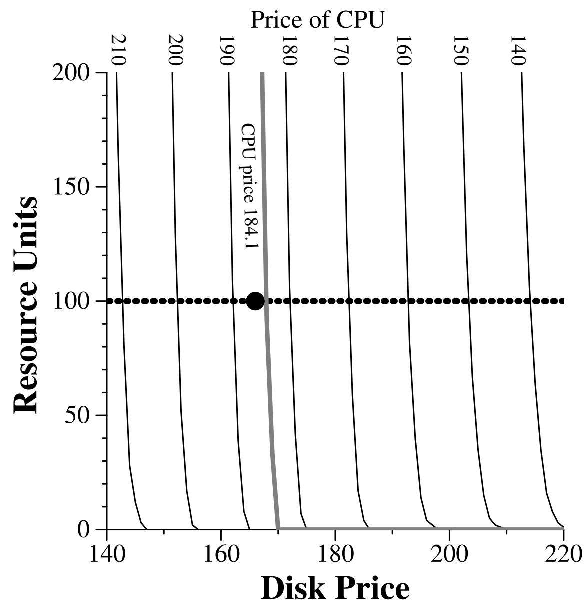

Figure 18. Disk aggregate supply and demand curves for the Bank of G, over-demand case, iteration 2000.

mate market-clearing solutions may fall. We also show the projection of this line of equilibria onto the price plane, and note that the price solutions indeed fall very near or upon this line of minima. Also important to note is that the projection is near linear with slope $= -1$. This serves as further confirmation that the two commodities are almost perfectly complementary. We conclude, based on this further evidence, that both our implementation of Smale's method and the First Bank of G are functioning correctly and achieving the results expected by the general theoretical formulation advanced by Smale as applied to our simple Grid economy. The results are particularly encouraging since they do not depend upon gross substitutability restrictions and because they can be achieved via an implementable system which does not require market-wide polling.

3.4 Revisiting under-demand

Having seen that our simulated economy converges to real equilibria in the overdemand case, we can re-examine the under-demand case again using our characterizations of its macroeconomic behavior. Figures 21 and 22 show the economic state of the simulation using Smale's method, iteration 3119. This timeslice occurs just after the beginning of a simulated "day", when jobs are injected into the system. The state of the system at this point is similar to the over-demand case, and this is reflected by the similarity of Figures 21 and 22 to Figures 15 and 16.

However, once the consumers' jobs for the day become serviced, the system enters an under-demanded state. Consumers get new jobs at an average rate of one every ten time steps, and they typically have plenty of \$G with which to service jobs. Producers on the other hand, are mostly idle. However, since they base their supply functions on average profit, they still refuse to sell until a certain threshold price is met. The state of the system during iteration 4000 is plotted in Figures 23 and 24, using the same linear scale for the y-axes as in the other graphs, and in Figures 25 and 26,

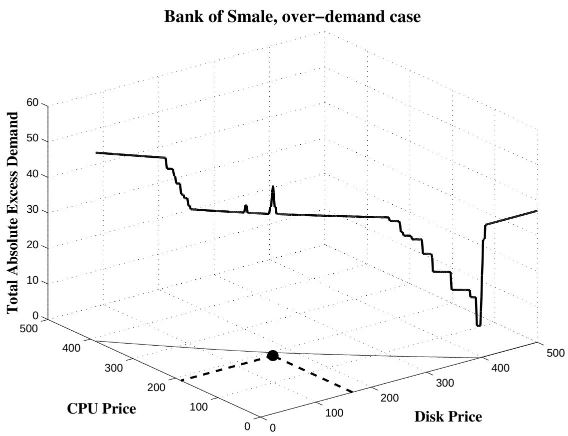

Figure 19. Total absolute excess demand minima, Smale's Method, overdemand case. The projection upon the price plane is also shown. Filed circles represent equilibrium price solutions at this iteration.

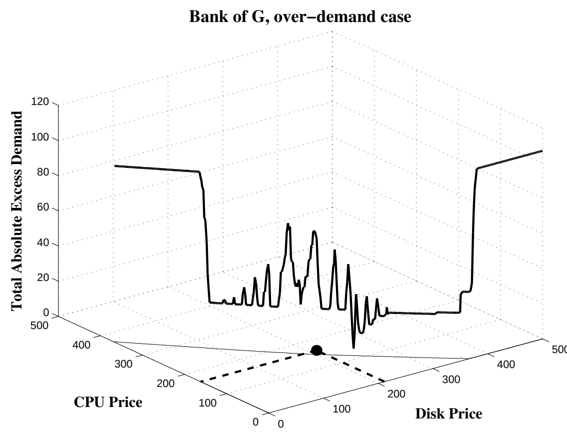

Figure 20. Total absolute excess demand minima, Bank of G, overdemand case.

using a more readable log scale.

Although it is difficult to discern from the figures, there is no equilibrium point for both commodities in this graph. This is because the system at this point is not a well-behaved economy, since the lowering of prices does not necessarily bring about an increase in demand. Put another way, the demand is so low that the assumption that individual agents do not make a significant difference is violated. Regardless, both Smale's method and the Bank of G default to a "normal" price. The market is not cleared – there is a supply glut – but prices do not become abnormally depressed. These results indicate that both Smale's method and the First Bank of G will be reasonably robust with respect to degeneration in the underlying economic behavior of the systems to which they are applied.

Probing further, the behavior of the banks in this case can be accounted for by looking at the supply and demand curves; note that the price that each bank finds is one where the supply curve is almost vertical and the demand curve horizontal, indicating a large jump in producer behavior at or near this price. This means that the excess demand function for each commodity will locally depend only on that commodity's price and will be extremely sensitive to small changes in price. Thus the Jacobian matrix $D_{\mathbf{z}}(\mathbf{p})$ will have the form

$$\begin{pmatrix} \text{very large negative number} & 0 \\ 0 & \text{very large negative number} \end{pmatrix}$$

The large diagonal entries will produce extremely small values of $\Delta\mathbf{p}$ for either price-adjustment scheme. Note in this case that Smale's method reduces to *tâtonnement* (*Cf.* Section 2.2.1) due to the off-diagonal zeros.

It is reasonable to expect that in more realistic simulations where true market behavior holds, and in any meaningful implementation of either of these price adjustment schemes, the behavior of the agents will be sufficiently heterogeneous as to preclude the existence of such large jumps in cumulative supply.

3.5 Efficiency

While commodities markets using Smale's method of price determination appear to offer bet-

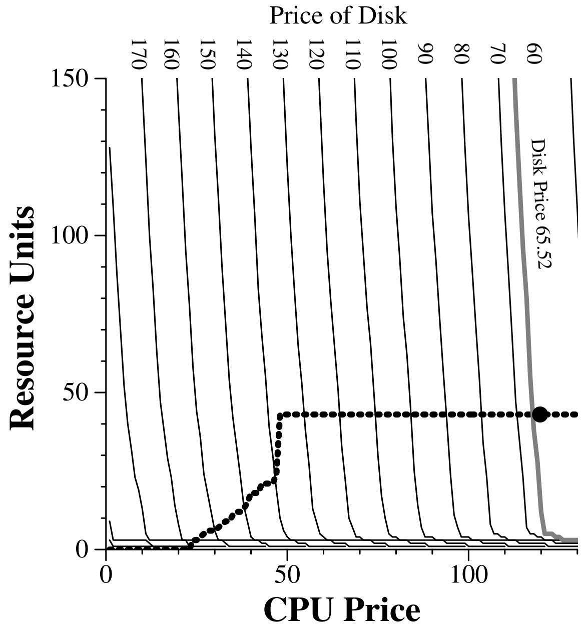

Figure 21. CPU aggregate supply and demand curves for Smale's method, under-demand case, iteration 3119.

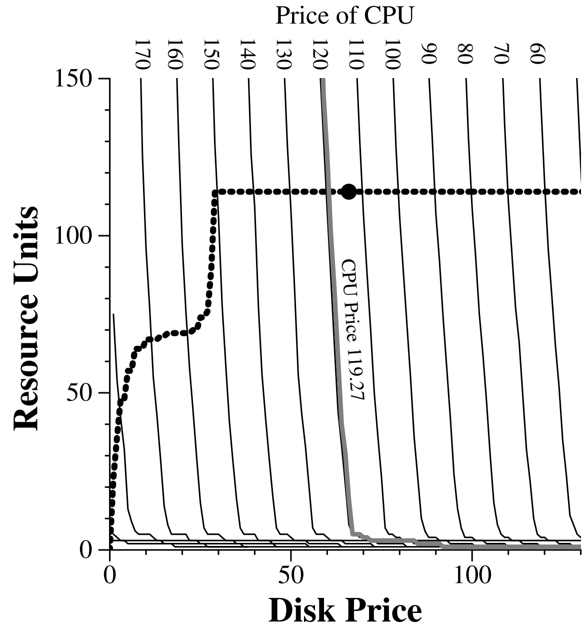

Figure 22. Disk aggregate supply and demand curves for Smale's method, under-demand case, iteration 3119.

ter theoretical and simulated economic properties (equilibrium and price stability) than auctions do, we also wish to consider the effect of the two pricing schemes on producer and consumer efficiency. To do so, we report the average percentage of time each resource is occupied as a utilization metric for suppliers, and the average number of jobs/minute each consumer was able to complete as a consumer metric. Table 2 summarizes these values for both the over- and under-demand cases.

In terms of efficiency, Smale's method is best and the First Bank of G achieves almost the same results. Both are significantly better than the auction in all metrics except disk utilization in the over-demanded case. Since CPUs are the scarce resource, disk price may fluctuate through a small range without consequence when lack of CPU supply throttles the system. The auction seems to achieve slightly better disk utilization under these conditions. In general, however, Smale's method and the First Bank of G approximation both outperform the auction in the simulated Grid setting.

4 Conclusions and Future Work

In this paper, we investigate G-commerce — computational economies for controlling resource allocation Computational Grid settings. We define hypothetical resource consumers (representing users and Grid-aware applications) and resource producers (representing resource owners who "sell" their resources to the Grid). While there are an infinite number of ways to represent individual resource supply and demand in simulated setting, and none are completely accurate, we have identified a set of traits that we believe are realistic.

- All entities except the market-maker act individually in their respective self-interests.
- Producers consider long-term profit and past performance when deciding to sell.

efficiency metric	under-demand	over-demand
Smale **consumer jobs/min**	0.14 j/m	0.05 j/m
B of G **consumer jobs/min**	0.13 j/m	0.04 j/m
auction consumer jobs/min	0.07 j/m	0.03 j/m
Smale **CPU** utilization %	60.7%	98.2%
B of G **CPU** utilization %	60.4%	93.9%
auction **CPU** utilization %	35.2%	85.5%
Smale **disk** utilization %	54.7%	88.3%
B of G **disk** utilization %	54.3%	84.6%
auction **disk** utilization %	37.6%	85.1%

Table 2. Consumer and Producer efficiencies

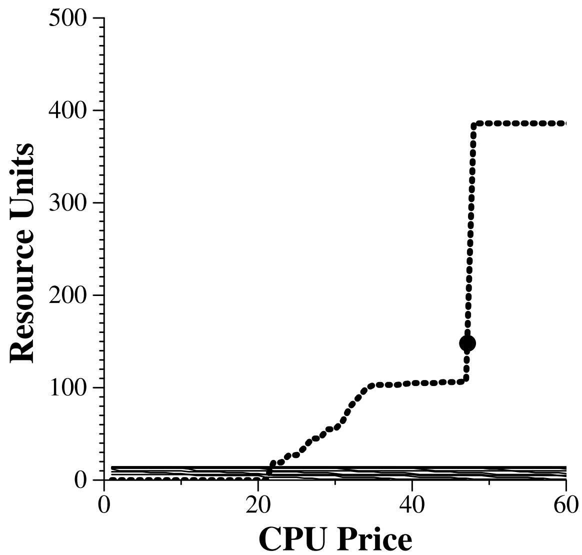

Figure 23. CPU aggregate supply and demand curves for Smale's method, under-demand case, iteration 4000.

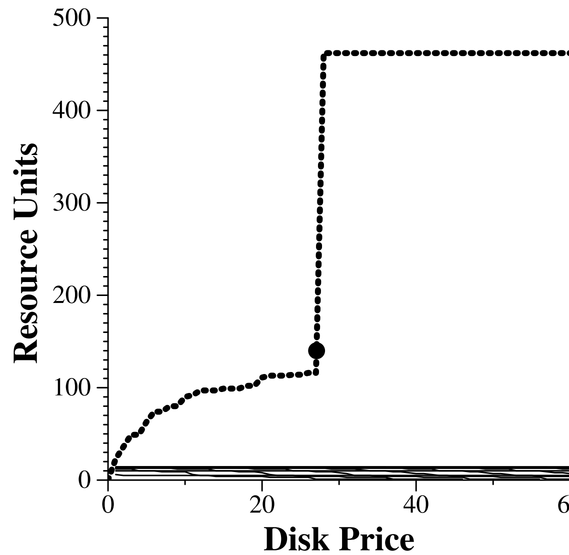

Figure 24. Disk aggregate supply and demand curves for Smale's method, under-demand case, iteration 4000.

- Consumers are given periodic budget replenishments and spend opportunistically.
- Consumers introduce work loads in bulk at the beginning of each simulated day, and randomly throughout the day.

Using simulated consumers and producers obeying these constraints, we investigate two market strategies for setting prices: commodities markets and auctions. Commodities markets are a natural choice given the fundamental tenets of the Grid [17]. Auctions, however, are simple to implement and widely studied. We are interested in which methodology is most appropriate for Grid settings. To investigate this question, we examine the overall price stability, market equilibrium, producer efficiency, and consumer efficiency achieved by three methods in simulation. The first implements the theoretical work of Smale [33] which describes how to ad-

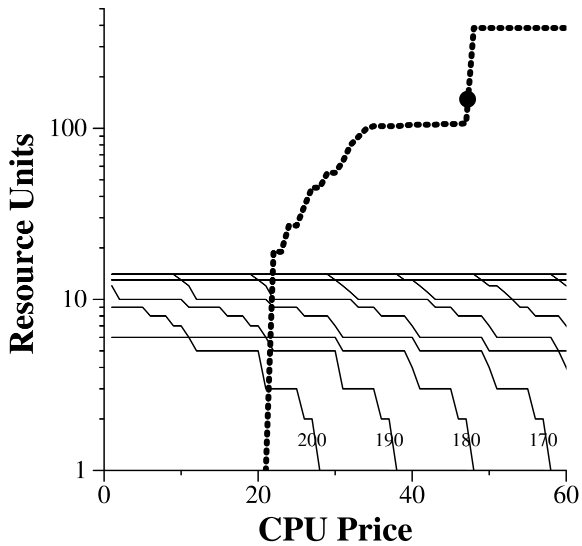

Figure 25. CPU aggregate supply and demand curves for Smale's method, under-demand case, iteration 4000, log y axis scale.

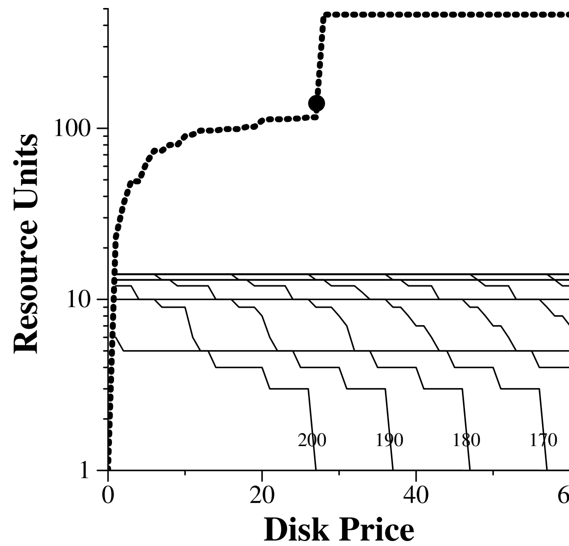

Figure 26. Disk aggregate supply and demand curves for Smale's method, under-demand case, iteration 4000, log y axis scale.

just prices in a commodities market to achieve equilibrium. It is viable in simulation, but impractical in the "real-world" as it relies on being able to poll reliably producers and consumers for supply and demand information. Often they do not know, or will not say what their response to a given price will be. The second method (The First Bank of G) is an implementable approximation to Smale's method. It uses a large-degree polynomial to approximate excess demand functions instead of polling making it parameterizable by observed market behavior only. Lastly, we simulate auctions in the style that has been investigated previously.

Our results show that Smale's results hold for our simulated Grid environment, despite badly behaved excess demand functions, and that the First Bank of G achieves results only slightly less desirable. In all cases, auctions are an inferior choice.

As part of our future work, we plan two parallel thrusts. First, we are exploring the space of plausible G-commerce formulations. Our goal is to identify and test, in simulation, different possible economies for the Grid. Secondly, we plan to construct a working version of the First Bank of G. Our previous work with the Network Weather Service [38, 39] and IBP [27] leaves us with the infrastructure necessary to build a large scale supply and demand information repository. Using the First Bank of G, we can generate prices based on "live" supply and demand information.

References

[1] D. Abramson, J. Giddy, I. Foster, and L. Kotler. High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid? In *Proceedings of the International Parallel and Distributed Processing Symposium*, May 2000.

[2] O. Arndt, B. Freisleben, T. Kielmann, and F. Thilo. Scheduling parallel applications in networks of mixed uniprocessor/multiprocessor workstations. In *Proceedings of ISCA 11th Conference on Parallel and Distributed Computing*, September 1998.

[3] F. Berman, A. Chien, K. Cooper, J. Dongarra, I. Foster, L. J. Dennis Gannon, K. Kennedy,

C. Kesselman, D. Reed, L. Torczon, , and R. Wolski. The grads project: Software support for high-level grid application development. Technical Report Rice COMPTR00-355, Rice University, February 2000.

[4] F. Berman, R. Wolski, S. Figueira, J. Schopf, and G. Shao. Application level scheduling on distributed heterogeneous networks. In *Proceedings of Supercomputing 1996*, 1996.

[5] J. Bredin, D. Kotz, and D. Rus. Market-based Resource Control for Mobile Agents. Technical Report PCS-TR97-326, Dartmouth College, Computer Science, Hanover, NH, Nov. 1997.

[6] J. Bredin, D. Kotz, and D. Rus. Market-based resource control for mobile agents. In *Second International Conference on Autonomous Agents*, pages 197–204. ACM Press, May 1998.

[7] J. Bredin, D. Kotz, and D. Rus. Utility driven mobile-agent scheduling. Technical Report PCS-TR98-331, Dartmouth College, Computer Science, Hanover, NH, October 1998.

[8] H. Casanova and J. Dongarra. NetSolve: A Network Server for Solving Computational Science Problems. *The International Journal of Supercomputer Applications and High Performance Computing*, 1997.

[9] H. Casanova, G. Obertelli, F. Bermand, and R. Wolski. The AppLeS Parameter Sweep Template: User-Level Middleware for the Grid. In *Proceedings of SC00*, November 2000. to appear.

[10] J. Q. Cheng and M. P. Wellman. The WALRAS algorithm: A convergent distributed implementation of general equilibrium outcomes. *Computational Economics*, 12:1–24, 1998.

[11] B. Chun and D. E. Culler. Market-based proportional resource sharing for clusters. Millenium Project Research Report, `http://www.cs.berkeley.edu/~bnc/papers/market.pdf`, Sep 1999.

[12] B. N. Chun and D. E. Culler. Market-based proportional resource sharing for clusters. Millenium Project Research Report, Sep. 1999.

[13] G. Debreu. *Theory of Value*. Yale University Press, 1959.

[14] B. C. Eaves. Homotopies for computation of fixed points. *Mathematical Programming*, 3:1–22, 1972.

[15] B. Ellickson. *Competitive Equilibrium: Theory and Applications.* Cambridge University Press, 1993.

[16] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications*, 1997.

[17] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, Inc., 1998.

[18] I. Foster, A. Roy, and L. Winkler. A quality of service architecture that combines resource reservation and application adaptation. In *Proceedings of TERENA Networking Conference*, 2000. to appear.

[19] J. Gehrinf and A. Reinfeld. Mars - a framework for minimizing the job execution time in a metacomputing environment. *Proceedings of Future general Computer Systems*, 1996.

[20] A. S. Grimshaw, W. A. Wulf, J. C. French, A. C. Weaver, and P. F. Reynolds. Legion: The next logical step toward a nationwide virtual computer. Technical Report CS-94-21, University of Virginia, 1994.

[21] J. M. M. Ferris, M. Mesnier. Neos and condor: Solving optimization problems over the internet. Technical Report ANL/MCS-P708-0398, Argonne National Laboratory, March 1998. `http://www-fp.mcs.anl.gov/otc/Guide/TechReports/index.html`.

[22] A. Mas-Colell. *The Theory of General Economic Equilibrium: A Differentiable Approach*. Cambridge University Press, 1985.

[23] O. H. Merrill. Applications and extensions of an algorithm that computes fixed points of certain upper semi-continuous point to set mappings. Ph.D. Dissertation, Dept. of Ind. Engineering, University of Michigan, 1972.

[24] H. Nakada, H. Takagi, S. Matsuoka, U. Nagashima, M. Sato, and S. Sekiguchi. Utilizing the metaserver architecture in the ninf global computing system. In *High-Performance Computing and Networking '98, LNCS 1401*, pages 607–616, 1998.

[25] N. Nisan, S. London, O. Regev, and N. Camiel. Globally distributed computation over the Internet — the POPCORN project. In *International Conference on Distributed Computing Systems*, 1998.

[26] N. N. Ori Regev. The popcorn market - an online market for computational resources. First International Conference On Information and Computation Economies. Charleston SC, 1998. To appear.

[27] J. Plank, M. Beck, and W. Elwasif. IBP: The internet backplane protocol. Technical Report UT-CS-99-426, University of Tennessee, 1999.

[28] B. Rajkumar. Ecogrid home page `http://www.csse.monash.edu.au/~rajkumar/ecogrid/index.html`.

[29] B. Rajkumar. economygrid home page `http://www.computingportals.org/projects/economyManager.xml.html`.

[30] H. Scarf. Some examples of global instability of the competitive equilibrium. *International Economic Review*, 1:157–172, 1960.

[31] H. Scarf. *The Computation of Economic Equilibria*. Yale University Press, 1973.

[32] S. Smale. Price adjustment and global Newton methods. *Frontiers of Quantitative Economics*, IIIA:191–205, 1975.

[33] S. Smale. Dynamics in general equilibrium theory. *American Economic Review*, 66(2):284–294, May 1976.

[34] T. Tannenbaum and M. Litzkow. The condor distributed processing system. *Dr. Dobbs Journal*, February 1995.

[35] C. A. Waldspurger, T. Hogg, B. Huberman, J. O. Kephart, and S. Stornetta. Spawn: A distributed computational economy. *IEEE Transactions on Software Engineering*, 18(2):103–117, February 1992.

[36] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta. Spawn: A distributed computational economy. *IEEE Trans. on Software Engineering*, 18(2):103–117, February 1992.

[37] L. Walras. *Eléments d'Economie Politique Pure*. Corbaz, 1874.

[38] R. Wolski. Dynamically forecasting network performance using the network weather service. *Cluster Computing*, 1998. also available from http://www.cs.utk.edu/ rich/publications/nws-tr.ps.gz.

[39] R. Wolski, N. Spring, and J. Hayes. The network weather service: A distributed resource performance fore casting service for metacomputing. *Future Generation Computer Systems*, 15(5-6):757–768, October 1999. available from `http://www.cs.utk.edu/~rich/publications/nws-arch.ps.gz`.