Chapter 4

Options and Commodity Markets for Computing Resources

Dan Cristian Marinescu, John Patrick Morrison, and Howard Jay Siegel

4.1. Introduction and Motivation

In this chapter, we introduce basic concepts regarding the role of commodity markets for computing resources supporting service-based computing and give an overview of Pleiades, an infrastructure for service-based computing. We discuss pricing strategies and the role of option trading in a commodity market for computing resources. Lastly, we review a macroeconomic model for computer services based on utility, satisfaction, and pricing.

The Internet and the World Wide Web have profoundly changed the way our society conducts its business; education, research, healthcare, economy, defense, and virtually all other human activities take advantage of the services and the information accessible through the Internet. The computing and communication infrastructure, which has enabled these changes, is now poised to take advantage of new technologies and it seems reasonable to expect that a service-based computing economy will soon become a reality.
Throughout this chapter the term *service-based computing* is used to mean a World Wide Web of domain-specific services offered by autonomous service providers. The purpose of a *service-based computing economy* is to provide the resources necessary to offer these services and, in this chapter, we advocate using a commodity market for the provision and acquisition of computing resources.

Market-oriented grids allow computing and communication resources to be traded; if they become a reality, options-based trading could play a role in grid computing. However, today’s grid computing is restricted to relatively few users with data-intensive applications requiring vast amounts of computing resources available only at a few sites. Also, grid resources are owned by a relatively small number of organizations. A commodity market for computing resources may have a greater impact upon a service-based computing economy than on grid computing, simply because it has the potential to reach a much larger audience.

A commodity market for software, hardware, information, and knowledge, coupled with an infrastructure for service-based computing, will support Web-based service providers by guaranteeing that they can call upon the resources needed to support their offerings. We believe that this resource provisioning will, in turn, stimulate the further growth of Web-based service delivery in many fields such as education, finance, and consulting. How grid computing will respond to the challenges posed by a service-based computing economy is a matter of speculation.

The scale of the systems considered here is considerably larger than that of current systems. We postulate a large number of sites with excess computing resources, and, a smaller, but still substantial, cohort of resource-starved service providers. In
considering where these resources will come from we note that the home computer of the future will most likely be a multi-core system with tens, if not hundreds, of cores; Gbytes of cache; tens of Gbytes of main memory; and hundreds of Gbytes, if not Terabytes, of secondary storage. These machines will be connected to the Internet via high-speed links and will allow owners to (remotely) control and monitor a wide range of appliances, including the home environment and security system. In addition to advances in the home computer, Web-based services are likely to proliferate; service-based computing will be integral to many applications, making them easy to use, and appealing to a very large user population.

The markets we envision will allow a very large number of owners of home computers to trade their excess resource capacity for a variety of incentives such as monetary rewards, service credits, and software maintenance contracts, all at a much larger scale than that envisioned for grid computing. It should be clear that sharing of home computer resources can happen only if:

(a) incentives for owner are substantial, and

(b) an infrastructure guaranteeing fast remote access and tight security is in place.

As software systems become increasingly more complex and the diversity of application software increases, the cost of software maintenance will dominate the computing costs for the majority of users. The prospect of remote diagnostics, automatic operating system updates, and assurance of compatibility between the operating system and application software may convince personal computer owners to join an infrastructure where these items are included in the incentive package.
Our approach is focused on the role that the options market could play in distributed systems, and, in particular, in a service-based computing economy. An option [14, 25, 27, 28] is a binding contract between, in our case, resource providers and resource seekers. Options on indices, futures, currencies, commodities, as well as equity options, are traded on all major exchanges. Options are derivatives because they derive their value from an underlying asset.

The concepts discussed here are consistent with the philosophy and the design principles of autonomic computing [15], the semantic Web [17], peer-to-peer computing, and economic models for large-scale distributed systems [19, 23, 24, 29, 30], but they push the envelope of these paradigms and break away from engrained ideas such as:

A home computer is a dedicated utility and that its local operating system should be in total control of its resources, including cores and CPU cycles, and primary and secondary storage.

- Contracting out the excess capacity of a home computer and allowing an external entity to control some of the local resources is socially and technically heretical. Note that even specific peer-to-peer systems, which use resources distributed over the Internet, are installed only with the owner’s agreement;

- If a service provider does not have total control of the resources necessary to support the service, it is not feasible to provide Quality of Service (QoS) or security guarantees or to ensure fault-tolerance.

Table 4.1 summarizes important attributes of peer-to-peer systems, autonomic computing, and Pleiades. The next section will address the question of how the Pleiades system will provide QoS and security guarantees while reducing the investments that
service providers have to make in hardware and software and, at the same time, reducing the software maintenance costs for the owners of home computers.

<table>
<thead>
<tr>
<th></th>
<th>Peer-to-peer Systems</th>
<th>Autonomic Computing</th>
<th>Pleiades</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce investments in hardware systems for service providers</td>
<td>Yes – uses resources scattered over the Internet</td>
<td>To some extent.</td>
<td>Yes – uses resources scattered over the Internet</td>
</tr>
<tr>
<td>Reduce software maintenance costs for service providers</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reduce software maintenance costs for home computer owners</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Provide Quality of Service (QoS) guarantees</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Provide security guarantees</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4.1. A comparison of several attributes of peer-to-peer systems, autonomic computing, and the planned Pleiades system.

The chapter has is organized as follows. In Section 4.2 we present the design objectives and the organization of the Pleiades system; the system can benefit from a market-oriented economy supporting option trading. Section 4.3 provides an overview of commodity markets for computing resources and pricing strategies. Then, in Sections 4.4 and 4.5 we introduce basic concepts regarding options and option pricing in financial markets. In Section 4.6 we discuss the role of options in a commodity market for computing resources. Finally, in Section 4.7 we provide an overview of a macroeconomic model for commodity markets for computing resources.
4.2. A Service-Oriented System Based upon Resource Virtualization

Traditionally, a service provider in a service-based distributed system owns, or is in total control of, all the resources needed to provide its service. Peer-to-peer systems are challenging this paradigm with varying degrees of success. For example, Skype, an Internet phone service provider, surreptitiously uses selected sites connected to the system as super nodes to route calls. Napster, a file sharing system stores data files on individual sites owned by subscribers. Video-on-demand is an application attempting to take advantage of data storage on sites scattered around the Internet.

The strategy used by these applications and others is motivated by the desire to reduce the cost of providing services. Current implementations of this resource sharing philosophy is plagued by problems such as uncertainty regarding the number of resources available at any instant in time, and the lack of system support at those sites needed to implement those services. As a result, such systems are unable to provide QoS guarantees or to ensure fairness, are vulnerable and prone to catastrophic failures, do not provide security guarantees either to the supplier of the resources or to the user of the system (often the same individual), are restricted to a few services with a simple user interface, and are limited in their ability to use a wide spectrum of resources. The lack of an enabling structure to mediate among suppliers and consumers of resources makes peer-to-peer systems scalable, but the disadvantages of the approach seem to be severely limiting. The next evolutionary step in service-based computing and communication systems could use a distributed infrastructure to mediate QoS and security guarantees, enable interested parties to negotiate reliability and fault-tolerance levels, and support graceful performance degradation on the occurrence of catastrophic events.
We consider a service-based computing economy where resources are distributed throughout the network \([7, 8, 9, 16, 18, 21, 31]\). The Pleiades system mirrors the realities of financial markets. The component we call the “Market Authority” buys computational resources from individuals, bundles them, and sells them to the providers of computing services; thus, service providers no longer need to make costly and often risky investments in very large computing facilities together with their associated high costs of ownership. This strategy is consistent with autonomic computing promoted by IBM, but emphasizes distributed resources rather than resources concentrated in a few sites.

We now examine briefly those attributes of a service-oriented system that we believe to be critical in addressing the problems mentioned above.

- **Self-organization.** A system is self-organizing if its components interact to dynamically achieve a global function or behavior \([12]\). Self-organizing systems typically have higher level properties that cannot be observed at the level of the individual components and that can be seen as a product of their interactions (they are more than the sum of their parts). Complex systems exhibit a high degree of self-organization. Recent observations of power-law distributions in the connectivity of complex networks came as a big surprise to researchers steeped in the tradition of random networks. Even more surprising was the discovery that power-law distributions also characterize many biological and social networks. Many attributed a deep significance to this fact, inferring the existence of a “universal architecture” of complex systems. Self-organization can be best explained in terms of the network model of a system \([6]\) where the vertices represent the principals and the links represent the relationships among them. Self-organization implies that the probability
that one node of a graph is connected with \( m \) other nodes, \( P(m) \), decays as a power law when a vertex interacts with \( m \) other vertices, \( P(m) \approx m^{-\gamma} \) [4] regardless of the type and function of the system, the identity of its constituents, or the relationships between them. This implies that large systems self-organize into a scale-free state when \( P(m) \) has the expression given above regardless of the number of nodes of the network [2, 3, 4]. The term scale-free, though linguistically questionable, is widely used in the literature to quantify scalability in the context of a random graph model of a system. Scale-free means that the probability that one node of the graph is connected with \( m \) other nodes, \( P(m) \), is independent of the scale of the network, i.e., the number, \( N \), of nodes in the graph. Empirical data available for social networks, power grids, the Web, and the citation index of scientific papers confirm this trend.

As an example of a social network, consider the collaborative graph of movie actors, where links are present if two actors were ever cast together in the same movie; in this case \( \gamma \approx 2.3 \). The power grid of the Western US has some 5000 vertices representing power generating stations and in this case \( \gamma \approx 4.0 \). For the World Wide Web, the exponent \( \gamma \approx 2.1 \); this means that the probability that \( m \) pages point to a particular page is \( P(m) \approx m^{-2.1} \). Recent studies indicate that for scientific papers; this means that the probability that \( m \) papers referring to the same paper is \( P(m) \sim m^{-3} \) [4]. The larger the network, the closer is the distribution approximated with a power law with \( \gamma = 3.0 \), \( P(m) \sim m^{-3} \). To be scalable and fair, and to take advantage of temporal and spatial locality of resources, services, and users, a service-oriented system should be self-organizing. Virtual-organizations. The members of a service oriented community should be able to form Virtual Organizations consisting of resource
providers, service providers, users of services, or any subset of these categories. In principle, Virtual Organizations can use the basic mechanisms provided by the system (e.g., contract negotiations protocols, performance evaluation tools, data transport mechanisms) to develop their own security, QoS, and fault-tolerance policies, and subsequently behave as autonomous entities – independent of the other communities using the service-oriented framework. Virtual Organizations may be created dynamically and exist only for the duration of a contracted period. The statement that the complexity of a system increases with the number of its elements, the number of interactions among them, the complexity of individual elements, and the complexity of the interactions is generally accepted [10, 13, 18]. To ensure scalability of our system, we devote particular attention to the algorithms and policies necessary for supporting the formation of Virtual Organizations. Our Virtual Organizations are nearly self-sufficient entities: trading outside the organization occurs only when the demand and supply cannot be satisfied within the organization. The members of a Virtual Organization agree to share resources such as CPU cycles, primary memory, secondary storage, software, information, and data in a dependable, flexible, economical, and secure manner subject to a set of individual constraints. We expect Virtual Organizations to have a number of benefits: including (a) improved scalability; (b) increased satisfaction levels for both resource suppliers and resource consumers, and support for better matching of contract requirements of suppliers and consumers; and (c) better utilization of resources and lower penalties on defaulted contracts. In general, contracted parties are rewarded for fulfilling contractual obligations and penalized otherwise; by continually meeting contractual obligations,
individuals can build up their reputation over time [20, 32]. Our preliminary studies indicate that these expectations are well founded. For example, Figure 4.1 shows the “average matching level” function of the population of agents (suppliers and consumers) based on a simulation study we are now conducting. The *average matching level* measures the absolute value of the difference of corresponding attributes of “call” and “put” contracts (described in Section 4.4) once they are matched together; the lower the value of this discrepancy the higher the average satisfaction level of the agents involved in the contract. As can be seen in Figure 4.1, the matching level improves when contracts are carried out by the local Market Authority in each Virtual Organization, referred to as VO-MA, rather than by the global Market Authority (the MA).
Figure 4.1. The average matching level as a function of the population size with and without Virtual Organizations, based on a simulation study. The average matching level measures the absolute value of the difference of corresponding attributes of “call” and “put” contracts once they are matched together; the lower the value of this discrepancy the higher the average satisfaction level of the agents involved in a contract. (a) When each Virtual Organization has its own Market Authority, VO-MA, the value of the matching level when contracts are carried out within a Virtual Organization is better than the case (b) when a global Market Authority is involved. 95% confidence intervals are shown.

- **Security and Trust Management.** We assume that resource providers, service providers, and users of services have different security constraints and policies. Thus,
our system has to provide basic mechanisms for enabling security and trust and to allow individuals to negotiate as appropriate. Security for resource providers metaphorically means that when they “cut a slice” of their resources and offer them to the pool of global resources, they are guaranteed that the functionality of their slice is maintained and no security issues are introduced. This is in stark contrast with existing systems such as the Grid, which requires third party software to be installed on local systems without guaranteeing that the functionality of the local system is maintained, or that the security of the local system is not compromised. Service providers and users of services have to trust that resources distributed throughout the system satisfy their security constraints. For example, a user of a service must trust the service provider to fulfill its obligations regarding security, QoS, and other user policies. In turn, the service provider must trust the suppliers of resources, and so on.

- **Resource Virtualization.** Resource virtualization allows transparent access to resources. In our context, this simply means that the resources available to a service provider are actually distributed throughout the system in a similar way that the pages of a virtual address space are distributed throughout the physical memory.

- **Incentives, Rewards, and Penalties.** Let us assume that we have in place an ideal system with all the ingredients mentioned above and that there is a very large population of potential resource suppliers who enjoy a substantial amount of excess resources such as CPU cycles, primary and secondary storage, and communication bandwidth. The missing ingredient is to motivate them to offer this excess local capacity for use by the wider community. This can be accomplished by a computing economy that provides incentives. Moreover, rewards should be in place to encourage
their continued participation. Of course, penalties should also be imposed when a contract is broken or anti-social behavior is detected.

Our Pleiades system is designed around a Market Authority (MA), a distributed infrastructure to mediate QoS and security guarantees that enable interested parties to negotiate reliability, and to provide fault-tolerance levels that support graceful performance degradation on the occurrence of catastrophic events. Pleiades, shown in Figure 4.2, allows service providers to acquire virtual resources supplied by a large user community willing to trade excess capacity for incentives such as service credits.

Pleiades is predicated on the assumption that resources are contracted to the MA, and the MA bundles these resources based on attributes such as quantity, price, QoS, security, location, and network access in an effort to match these attributes to the needs of the resource consumer community. Though immaterial for the present discussion, we note that the resource consumers may also be providers of services that may, in turn, be requested by some of the resource providers. For example, in Figure 4.2, entity A accesses services provided by 1 and 2; service providers 1 and 2 use resources provided by B and C.

The basic organization of a service-provider is illustrated in Figure 4.3. The critical component is a QoS-aware and fault-tolerant Standard Virtual Machine (SVM). The term “standard” reflects the fact that SVMs issue from the MA.
Figure 4.2. Conceptual organization of the Pleiades architecture. The Market Authority contracts with individual resource suppliers (labeled A, B, C) access to local resources and bundles these resources as needed by service providers (labeled 1, 2, 3). Clients request services, e.g., A accesses services provided by 1 and 2; service providers 1 and 2 use resources provided by B and C. Virtual Organizations mimic the layout in this figure; they have their own Market Authority.
Figure 4.3. A service provider runs application specific software; a dispatcher uses the resources provided by a Standard Virtual Machine (SVM) that interacts with the Market Authority and Guest Partitions (GPs) (see Figure 4.4), running at many sites.
We now discuss the hierarchy of Virtual Machines (VMs) running on future home computers containing tens or hundreds of cores. The cores as well as the other system resources are split among: (a) A Control Partition (CP); (b) A Host Partition (HP); and (c) A Guest Partition (GP), as shown in Figure 4.4. The First Level Virtual Machine (FLVM) is a tamper-proof software component, obtained from the MA, which periodically certifies its compliance to system-wide security standards. Once an FLVM is installed on a local machine, the owner can initiate contract negotiations for the supply of a portion of the local resources and in return for credit that can be used to purchase various services or can be en-cashed for financial recompense. For the sake of simplicity, we assume that the excess resources are negotiated as a single package with the MA. When a negotiation is completed, the FLVM provides a “Capability List” to the Second Level Virtual Machines (SLVMs) that control resource allocation in the Host and Guest Partitions. To ensure fault-tolerance, the FLVM and SLVM are replicated and run on multiple cores. Every FLVM has a unique FLVMId, and at creation time is provided with the information necessary to access the MA. Once a contract is negotiated and agreed between the MA and the Host, the FLVM loses control of the local resources subject to that contract. An FLVM can regain control of these resources again only after a renegotiation process, which, in turn, can only happen with the approval of the MA.
Figure 4.4. The local system configuration of a resource provider. The cores are shared among three partitions: Control, Host, and Guest. The First Level Virtual Machine (FLVM) controls the resource partitioning and enforces security policies and contracts. The FLVM is certified by the Market Authority and it is replicated on multiple cores to ensure fault-tolerance. The Host Partition is controlled by a Second Level Virtual Machine (SLVM) and is also replicated on several cores. Each core in this partition is controlled by a Third Level Virtual Machine (TLVM) that can load an Operating System (OS), or another Control Program (CP) and run an application. The Guest Partition also is controlled by an SLVM and cores in this partition run a service under the control of a CP.

The process of bundling resources is called resource virtualization. Pleiades’ resource virtualization requires special features of the hardware and software that are largely unavailable at present:
1. The strict security model is based upon a hierarchy of virtual machines running on individual nodes, supplying resources to a common pool. The FLVM should ensure an impenetrable security wall between the local resources contracted by the Host, the owner of a home computer, and the Market Authority. This is in stark contrast with existing operating systems which vest total control in local resources.

2. The hardware support for security should include a memory architecture based on multiple virtual spaces, and multiple virtual processors that can be allocated dynamically. A segmented memory, multi-core context switching, and access key controlled I/O are some of the hardware features required to support strict partitioning of local resources.

3. Different functional components of the Market Authority perform critical functions for the Pleiades system such as:

   (a) Supplying and certifying critical software components: the SVMs for service providers, Figure 4.3, and the FLVMs for resource providers, Figure 4.4.

   (b) Conducting contract negotiations between parties: conducting contract renegotiations subject to the approval of all parties involved; monitoring strict enforcement of existing contracts; gathering data relating to the reputation of the parties involved; and disseminating reputation information.

   (c) “Bundling” resources from multiple suppliers so as to match the requirements of service providers.
(d) Ensuring fair allocation of resources to service providers.
(e) Ensuring fair compensation of resource suppliers.
(f) Enforcing measures to achieve graceful performance degradation when available resources fall below a “low water mark,” and enforcing system stability through market interventions in the case of unexpected surges of load or catastrophic resource failures.

4. Complex negotiation protocols allowing individual service providers to specify QoS requirements as well as support for “options-based” contracts.

5. QoS-aware and fault-tolerant robust [1, 7, 26] resource management system embedded into SVMs running at the site of individual service providers. SVMs are customizable system components that enforce a level of service dictated by the specifications of the service provider and consistent with the reputation of each resource supplier in the “bundle” the service provider gets from the Market Authority.

6. Coordination engines allowing users to combine a wide range of both fine-grain and course-grain services. A fine-grain service typically requires low-level resource consumption, have a simple and well-defined interface, and a very short response time. For example, some of the services exposed by Google map, and used for mashups, would qualify as fine-grained services. A mashup is a Web application that combines data from more than one source into a new and distinct Web service that was not originally provided by either source. A coarse-grain service typically requires not only computing but also other types of resources,
has a complex interface, and persists for an extended periods of time. For example, a service requiring the manipulation of a network-based instrument (e.g., an electron microscope) qualifies as a coarse-grain service.

7. A versatile abstract machine to express the functions of the hierarchy of Virtual Machines as required in Pleiades, as shown in Figure 4.4.

A catastrophic system failure, detected either by a service provider or by the Market Authority, results in penalties. The FLVM maintains a log of critical events and can avoid penalties if failure is due to causes outside local control, e.g., power or network failures. Each core of a partition runs under the control of a Third Level Virtual Machine (TLVM) that may load an operating system or a control program on that core and execute an application.

The system presented in this section relies on a commodity market for computing resources and the next sections introduce basic concepts regarding commodity markets and option trading in financial markets.

4.3. Markets and Commodity Pricing

A market is a meeting place for: (a) investors who have excess resources, e.g., cash in case of financial markets; (b) organizations that need additional resources, e.g., companies that issue shares and use the revenue obtained by selling the shares to introduce new technologies, hire more personnel, or increase production capacity; and (c) organizations who mediate the transactions, e.g., exchanges and financial institutions,
Investors buy and sell securities (such as stocks, bonds, mutual funds, or futures), commodities (such as oil, precious metals, or agricultural goods), currency, and other valuable items; the prices of the items traded on financial markets reflect the market hypothesis. Efficient market hypothesis (EMH) assert that prices on traded assets, e.g., stocks, bonds, or property, reflect the collective beliefs of all investors about future prospects.

Computing and communication resources such as CPU cycles and communication bandwidth are perishable commodities that also can be traded. Paradoxically, recent advances in microprocessors, memory, storage, and communication technology, coupled with widespread Internet access, while significantly increasing the compute resources of the average owner, provide an impetus for the development of a distributed market economy trading these resource, because the need for computing resources among certain sectors (such as service providers) is, and will remain, insatiable.

In this section, we review classical pricing strategies from traditional markets and their applicability to service-based computing [14, 25, 27, 28]. We also discuss the relation between quality and price.

*Premium pricing* is the application of higher prices when there is uniqueness about the service and a competitive advantage exists. For example, access to a parallel system with a new and very fast interconnection network will be priced higher than a similar system with a more modest interconnection network. Similarly, if a service provides additional guarantees, e.g., guarantees on the response time, it will command a higher price.
Penetrating pricing is the application of reduced pricing to gain market shares, and once the goal is achieved the price increases. A new service may use this strategy to attract customers.

Economy pricing is the application of reduced pricing to dispose of abundant commodities or services. For example, if a service provider in the Pleiades system has a number of futures contracts cancelled, under favorable circumstances, it may decide to use economy pricing. Recall that the service provider is a consumer of resources; it “buys” resources it deems necessary and sufficient for the delivery of its services; in addition, it may buy options (see Section 4.4) to guard against the possibility that more resources then initially anticipated are needed.

Price skimming is applied when one has a competitive advantage, but this advantage is somehow not sustainable. Price skimming is a pricing strategy in which a marketer initially sets a relatively high price for a product or service and then lowers the price over time. This allows the supplier to recover its costs quickly before competition steps in and lowers the market price. This policy is effective in situations where a firm has a substantial lead over the competition with a new product or when there is a shortage of services/products. For example, in the Pleiades system a service provider may take advantage of power outages in some regions of the country to increase the price of its services – reflecting increased demand.

Product line pricing is applied when there is a range of services/products and the price reflects the benefits of parts of the range. A line can comprise related
products of various sizes, types, colors, qualities, or prices. Product line pricing is based on the correlations between the demand for the products of a line, the costs of the products, quality, etc. For example, a car maker usually has a line of cars and the prices may not reflect the actual costs; profit margins may be smaller for small cars, reflecting a larger volume of sales.

_Captive product pricing_ increases the price when the service/product has attributes that force a customer to use a particular provider. For example, once one installs an operating system on a home computer, application software such as browsers from other software companies may not work well or may not run at all. In the context of the computing services, when the format of the results produced by one service is non-standard, then the provider of that service may charge a premium price for another service capable of further processing the results.

_Service bundle pricing_ combines multiple services in the same package and charges a smaller price than the sum of the prices charged for individual services; this strategy forces the customer to pay for all services, regardless of use. For example, a cable service provider bundles several channels.

The concepts from financial markets cannot be applied directly to a commodity market for computing resources for a number of reasons. First, some of the computing resources, such as CPU cycles and bandwidth, are perishable and this will affect their price; if at time \( t_0 \) a contract for a resource that becomes available at time \( t_a \) is posted, then its price will typically decline slowly at first and then faster as the time approaches
$t_o$. Second, computing resources have attributes that do not exist in traditional commodity markets. An appliance will not distinguish a kilowatt of electricity produced by a windmill from one produced by a nuclear power plant; the electricity supplier can switch from one source to another and mix multiple sources of electricity at will without affecting any appliance connected to the electric grid. However, computing resources may not be interchangeable, limiting how they can be mixed or substituted transparently; such a substitution requires that some action be carried out at the point of consumption and thus leads to additional overhead. In our system, the MA offers CPU cycles from multiple sources, say, to a service provider, bundled together in the same package. If one of the sources fails, then the MA has to first find another source and then reduce the price per unit of service to compensate for the overhead experienced by the SVMs running at the service provider’s site to move data around and possibly to redo some computations at the new site. Moreover, the price of several resources such as CPU cycles and main memory when bundled together should reflect their joint utility to the customer.

In Section 4.7, we discuss two more pricing strategies used to encourage or to discourage consumption. To encourage consumption, the unit price is decreased as consumption increases; for example, for the first 1,000 kWh of electricity a consumer may be charged 8 cents/kWh, for the next 1,000, the price could be 7 cents/kWh, and when the consumption exceeds 2,000 kWh, a price of 5 cents/kWh is charged. To discourage consumption, the price per unit could be increased as the consumption increases.
A service provider in the Pleiades system can benefit from contracts that give the right, but not the obligation, to acquire resources. Such contracts enable a service provider to minimize its costs while being able to adapt to sudden change of demand for services. Financial markets support instruments, called “options,” the subject of the next sections.

### 4.4. Options - Basic Concepts

An *option* is a financial contract between two or more parties. The value of this contract is determined by fluctuations in some underlying asset (or assets). In its simplest form, an option is sold by one party (who becomes known as the option *writer*) to another party (who then becomes known as the option *holder*). The contract offers the holder the right, but not the obligation, to buy or to sell (in finance speak: *to call* or *to put*) an asset within a specified period of time for a price that is pre-agreed at the time the contract is made (this price is called the *strike price*).

As part of the contract, the writer is paid a sum of money by the holder. This *fee* obliges the writer to honor the contract (that is, to buy or sell the underlying asset) if the holder decides to exercise his option.

An option writer may *sell* either a *call option* or a *put option* (or both, of course). In selling a *call option*, the writer agrees to *sell* the asset to the option holder within the terms of the contract. Likewise, in selling a *put option*, the writer agrees to *buy* the asset from the option holder, also subject to the terms of the contract. (Reciprocally, an option holder *buys* call options and put options – enabling the buyer to buy or sell, at the buyer’s
discretion within the terms of the contract.) Option trading is a zero-sum game; the option buyer’s grain is the option seller’s loss and visa versa.

Options can be used to speculate; an option writer selling a call option is betting that the asset value will fall substantially relative to the strike price during the life of the option. In that case, the holder will not require the writer to sell the asset (why buy from the writer for a strike price that is higher than the current value of the asset?). Thus, the net gain to the writer (and the net loss to the holder) is the fee paid by the holder. Alternatively, if that asset value were to rise substantially relative to the strike price during the life of the option, the holder may require the writer to sell the asset for the strike price. After purchase, the holder can immediately sell the asset for a profit at the higher market value. The net gain for the holder will be the current value of the asset minus the sum of the option fee and the strike price. Similarly, the net loss to the writer will be the difference between the value of the asset plus the fee, minus the strike price.

In a similar manner, put options lead to speculative trading. The holder of a put option (which would force the writer to buy) is guarding against the value of the underlying asset falling. If that happens the holder is able to sell the asset for the (higher) strike price – thus minimizing potential losses. Conversely, the writer in this scenario is anticipating an increase in the asset value. Both cannot be right and a similar zero sum calculation, as in the call option scenario, pertains.

As a simple example, consider the following. On November 30, 2007 an investor decides to buy 1,000 IBM options to purchase stock with the expiration date of July 18, 2008 for the strike price of $120/share (the investor thus becomes a holder). At the time of purchase, the closing price of this stock is $105.18/share. The fee for this option at
$4.40/share, is 1,000 x $4.40/share = $4,400. This outlay compares favorably with 1000 shares x $105.18/share = $105,180, which would be the cost of buying 1,000 shares outright on November 30, 2007.

Now, assume that on March 4, 2008 the IBM stock trades at $143/share and the investor (holder) decides to exercise the option and pays the strike price ($120,000) for the 1,000 IBM shares (1000 shares x $120/share). These shares could be resold immediately at market value to yield $143,000 (1000 shares x $120/share). The investor’s (holder’s) profit, and indeed the writer’s loss, is thus 143,000 – (120,000 + 4,400) = $18,600.

Instead of selling, suppose the investor (holder) expects the stock to rise further and does not exercise the option on March 4, 2008 and the next day the stock plunges and trades at $110, remaining there until the option expires. In this case it makes no sense for the investor to exercise the option at the strike price of $120 and will loose the $4,400 option fee.

The writer of an option is a financial institution; underwriting options involves risks due to market volatility. In our example, assume that a financial institution underwrites 1000 call options with the strike price $120/share when the call option fee is $4.50/share and 1000 put options with the same strike price and a fee of $16.35/share. Thus, the writer collects $20,850 (4,500 +16,350) at the time when it underwrites the two batches of options. If, at the exercise date, the market is down and the price of a share is $100, then the call option will not be exercised, but the put option will be. Thus, the writer has to buy the 1000 shares from the holder of the put option and pay $120,000 for them. The writer can then resell the 1000 shares for only $100,000; as a result of these
transactions the writer makes a profit of $850 (20,850 - (120,000-100,000)). If the market goes up and the price at the exercise date is $145/share, then the holder of the put option has no reason to sell, but the holder of the call option will exercise his option and pay the writer $120,000. The writer will get the 1000 shares at the current market price and pay for them $145,000. This time the writer will have a loss of $4,150 (20,850 -(145,000-120,000)). Recall that the bid price is the highest price an investor is willing to pay to buy an option; the ask price is the lowest price a trader will accept to sell an option. In our calculations we used an average of the bid and strike prices.

Table 4.2 presents the information regarding the price of options as it appears in newspapers, or is provided by financial institutions. Specifically, it shows the call and put option prices in US dollars as a function of the strike price for IBM options for July 18, 2008 as of December 1, 2007. As we can see, for the call option the lower the strike price the larger the option price; indeed when the current price is around $105, the option to buy the stock for $65/share on, or before July 18, 2008 should cost more. The situation is reversed for the put options. If you want to sell of an IBM share at the strike price of $65/share or before July 18, 2008 then the option should cost you next to nothing. Such tables for different exercise dates show that the price of an option increases with the duration of the contract due to market volatility; it is harder to predict the range of the price of a stock a year from now than the next day’s price.

This section gave an overview of the options terminology and mechanism for financial markets. In the next section, we will examine this more rigorously.
<table>
<thead>
<tr>
<th>Strike price</th>
<th>Bid price (call option)</th>
<th>Ask price (call option)</th>
<th>Bid price (put option)</th>
<th>Ask price (put option)</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>42.40</td>
<td>43.00</td>
<td>0.45</td>
<td>0.55</td>
</tr>
<tr>
<td>70</td>
<td>37.80</td>
<td>38.30</td>
<td>0.70</td>
<td>0.85</td>
</tr>
<tr>
<td>75</td>
<td>33.50</td>
<td>33.70</td>
<td>1.05</td>
<td>1.20</td>
</tr>
<tr>
<td>80</td>
<td>29.10</td>
<td>29.30</td>
<td>1.55</td>
<td>1.65</td>
</tr>
<tr>
<td>85</td>
<td>24.90</td>
<td>25.20</td>
<td>2.20</td>
<td>2.30</td>
</tr>
<tr>
<td>90</td>
<td>20.90</td>
<td>21.10</td>
<td>3.10</td>
<td>3.20</td>
</tr>
<tr>
<td>95</td>
<td>17.20</td>
<td>17.40</td>
<td>4.20</td>
<td>4.40</td>
</tr>
<tr>
<td>100</td>
<td>13.80</td>
<td>14.10</td>
<td>5.80</td>
<td>6.00</td>
</tr>
<tr>
<td>105</td>
<td>10.90</td>
<td>11.10</td>
<td>7.70</td>
<td>7.90</td>
</tr>
<tr>
<td>110</td>
<td>8.30</td>
<td>8.50</td>
<td>10.10</td>
<td>10.30</td>
</tr>
<tr>
<td>115</td>
<td>6.20</td>
<td>6.30</td>
<td>12.90</td>
<td>13.10</td>
</tr>
<tr>
<td>120</td>
<td>4.40</td>
<td>4.60</td>
<td>16.20</td>
<td>16.50</td>
</tr>
<tr>
<td>125</td>
<td>3.10</td>
<td>3.30</td>
<td>20.00</td>
<td>20.20</td>
</tr>
<tr>
<td>130</td>
<td>2.15</td>
<td>2.30</td>
<td>24.10</td>
<td>24.40</td>
</tr>
<tr>
<td>135</td>
<td>1.45</td>
<td>1.55</td>
<td>28.60</td>
<td>29.90</td>
</tr>
<tr>
<td>140</td>
<td>0.95</td>
<td>1.05</td>
<td>33.30</td>
<td>33.90</td>
</tr>
<tr>
<td>145</td>
<td>0.60</td>
<td>0.70</td>
<td>38.30</td>
<td>38.90</td>
</tr>
</tbody>
</table>

Table 4.2. The *call* and *put* option prices in US $ as a function of the strike price for IBM options for July 18, 2008 as of December 1, 2007; the expiration date is the third Friday of the month, in this case, Friday July 18, 2008. The closing price on November 30, 2007 was $105.18/share. The strike prices are listed in increments of $5 and range from about 65% to about 160% of the closing price. The bid and ask prices for call options are on the left, and for put options on the right. The information is from Wachovia.
4.5. Option Pricing on Financial Markets and the Black-Scholes Model

The prices of assets are governed by the so-called efficient market hypothesis that state that: (a) the current price reflects the history and does not provide any additional information; and (b) markets respond immediately to new information about an asset [11, 14]. The asset price obeys the stochastic differential equation:

\[
\frac{dS}{S} = \sigma(dX) + \mu(dt) \quad (1)
\]

Here, \( S \) is the price of the asset at time \( t \), \( dS \) the change of the price in a short interval of time \( dt \), \( dX \) is a sample from a normal distribution, \( \sigma \), the volatility, reflects the standard deviation of the return, and \( \mu \) is the drift - an average rate of growth of the asset price. We assume that the asset volatility is a known function of time. This equation describes a stochastic process called a random walk. The solutions of this equation fit well with the time series for equities and indices gathered from observations of the markets, while the data regarding currency does not. In addition, the model does not predict large variations of the asset prices experienced sometimes by real markets.

Before proceeding we introduce the lognormal distribution used to model variables that are multiplicative products of a large number of independent factors, each one with a small contribution. A random variable whose logarithm is normally distributed is said to have a lognormal distribution; if \( Y \) is a random variable with a normal distribution then \( X = \exp(Y) \) has a lognormal distribution.
Figure 4.5. (a) The probability density function of a lognormal distribution with the mean and the standard deviation of the logarithm of the random variable $\mu$ and $\sigma$ (in the range $1/8 - 10$). (b) The cumulative distribution function.
The probability density function (pdf) and the cumulative distribution function (cdf) of a lognormal distribution with \( \mu \) and \( \sigma \) the mean and the standard deviation of the logarithm of the random variable are displayed in Figure 4.5 for a range of values of the mean and standard deviations; their analytical expressions are respectively:

\[
f(x; \mu, \sigma) = \frac{e^{-(\ln x - \mu)^2 / 2\sigma^2}}{x\sigma\sqrt{2\pi}} \quad \text{and} \quad F(x; \mu, \sigma) = \frac{1}{2} + \frac{1}{2}\text{erf}\left[\frac{\ln(x) - \mu}{\sigma\sqrt{2}}\right]
\]

(2)

Equation 1 describes a Markov process, the asset’s price depends solely on the current price and not upon the past history. When \( \mu \) is constant, and the price at time \( t_0 \) is \( S_0 \) then:

\[
S = S_0 e^{\mu(t-t_0)}
\]

(3)

The probability density function of \( S \) follows the lognormal distribution:

\[
\frac{1}{\sigma S \sqrt{2\pi}} e^{-\left(\log(S)/S_0 - (\mu - 1/2\sigma^2)t\right)^2 / 2\sigma^2t}
\]

(4)

and the stochastic process is called a lognormal random walk.

Equation 1, formalizes a basic tenet of financial theories, namely that there are never opportunities to make an instantaneous and risk-free profit. Virtually all financial models assume the existence of risk-free investments with a guaranteed return, such as a sound bank, or a government issued bond; an investor could make a risk-profit over time by depositing the cash in a bank, or by investing in bonds, but we should emphasize that this profit can only be made over time, it is not instantaneous. One can probably beat the
risk-free investment by investing in equities but, a greater return, implies a greater risk. There is no free-meal, instantaneous and risk-free profits are not possible. This brings us to the concept of arbitrage, the practice of taking advantage of a price differential between two or more markets. An arbitrager is an individual who seeks out and takes advantage of mispricing. If market prices do not allow for profitable arbitrage, prices are said to constitute an arbitrage equilibrium or arbitrage-free market. An arbitrage equilibrium is a precondition for a general economic equilibrium. The assumption that there is no arbitrage is used to calculate a unique risk neutral price for derivatives.

To discuss option pricing we denote by: \( V(S,t) \) the value of an option, \( C(S,t) \) the value of a call option, \( P(S,t) \) the value of a put option, \( E \) the exercise price, \( T \) the expiry date, and \( r \) the interest rate. We assume that the short-term interest rates are a known function of time. The value of the call and the put options at the expiry are called the payoff diagrams and are given respectively by the following expressions:

\[
C(S,T) = \max(S - E, 0) \quad \text{and} \quad P(S,T) = \max(S - E, 0)
\]

(5)

The payoff diagrams and the call and put options, function of the asset value, are shown in Figure 4.6.

Now we consider a portfolio consisting of an asset, a call and a put option with the same expiry date and the same exercise price, \( E \). The value of the portfolio is:

\[
\Pi = S + P - C
\]

(6)

where \( S \) is the price of the asset, \( C \) the value of the call option, and \( P \) the value of the put option. The payoff at the expiry is \( S + \max(E - S, 0) - \max(S - E, 0) \). When \( S \leq E \) the value of the portfolio is: \( S + (E - S) - 0 = E \); if, on the other hand, \( S \geq E \) the value of the
portfolio is: \( S + 0 - (S - E) = E \). We conclude that the payoff is always the same, the exercise price. If we assume the existence of a risk-free interest rate over the lifetime of the option and ask the question how much should an investor pay for a portfolio that gives a guaranteed return \( E \) at time \( t - T \) then the current value of the portfolio should be:

\[
S + P - C = E e^{-r(T-t)} \tag{7}
\]

This relationship between the underlying asset and its options is called the put-call parity. A simple argument shows that the return from this portfolio is the same as the return from a bank deposit; if this would not be the case, then an arbitrator could make an instantaneous profit by buying and selling options and shares, while borrowing and lending money from the bank.

The Black-Scholes partial differential equation is a fundamental result in computational financing \([28]\); it relates the value of an option, the price of the asset, the volatility, and the risk-free interest rate:

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0 \tag{8}
\]

where \( V(S,t) \) is either a call or a put option. Several assumptions are necessary to derive this equation:

1. The value of an option \( V(S,t) \) is only a function of the price of the asset, \( S \), and the time, \( t \).
2. The price \( S \) of the asset is based upon a lognormal random walk.
3. The assets are divisible.
4. The assets do not pay any dividends during the lifetime of the option.
5. The risk-free interest rate, \( r \), and the volatility \( \sigma \) are known function of time over the lifetime of the option.

6. There are no transaction fees for hedging a portfolio.

7. Selling short is permissible.

8. Trading the underlying asset is continuous.

Figure 4.6. (a) The payoff diagram (thick line) for a call option and the option value prior to expiry (thin line) function of the asset value \( S \). (b) The payoff diagram (thick line) for a put option and the option value prior to expiry (thin line) function of \( S \). The exercise price is \( E \).

The Black-Scholes equation is based on Itô’s lemma [28]; if \( S \) is described by a logonormal random walk thus, it satisfies equation (1), and if \( f(S) \) is a smooth function of \( S \) then:

\[
\frac{df}{dS} = \frac{df}{dS} (\sigma S dX + \mu S dt) + \frac{1}{2} \sigma^2 S^2 \frac{d^2 f}{dS^2} dt = \sigma S \frac{df}{dS} dX + (\mu S \frac{df}{dS} + \frac{1}{2} \sigma^2 S^2 \frac{d^2 f}{dS^2}) dt \quad (9)
\]

Itô’s lemma allows us to write:
\[ dV = \alpha S \frac{\partial V}{\partial S} dX + (\mu S \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \frac{\partial V}{\partial t}) dt \] (10)

It is easy to determine the boundary conditions for solving this partial differential equation for European options [27]. But we are primarily interested in American options; in this case we have a free boundary problem. At time \( t \) there is a particular value of the asset value \( S \), called the optimal exercise price and denoted as \( S_f(t) \), that marks the boundary between two regions: in one region we should hold the option, and in the other region we should exercise the option. The valuation of American options demands several constraints:

- The option value must be greater than or equal to the payoff function.
- The option value must be a continuous function of \( S \).
- The first derivative (the slope) of the option value must be continuous.
- The Black-Scholes equation should be replaced by an inequality.

When we assume that the return from the portfolio cannot be greater than the return from a bank deposit, then we can show that for an American put option the following inequality should hold [28]:

\[ \frac{\partial P}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rS \frac{\partial P}{\partial S} - rP \leq 0 \] (11)

This section explored some of the mathematics underlying the financial commodity market. In the next section we discuss commodity markets for computing resources.
4.6. Option Pricing in Commodity Markets for Computer Resources

As mentioned earlier, options are critical for a commodity market of computing resources but striking a price for options poses many challenges. Moreover, traditional concepts from financial markets cannot be directly applied to determining prices for computing assets or associated options.

Options in a commodity market for computing resources are likely to play a more intricate role than they have in financial markets for a variety of reasons. Options are likely to be a critical component of such a commodity market because we cannot possibly ensure critical attributes of the system such as Quality of Service (QoS), fault-tolerance, or security, when all resources are traded on the spot-market (that is, the cash market or physical market). Long-term contracts and overprovisioning are necessary, but not sufficient; service providers should have the flexibility offered by the right, but not the obligation, to acquire resources when they are certain that those resources will be needed. Another major difference between the financial markets with a few, well-established exchanges, and computer resource commodity markets is due to the need for self-organization; quality of service, fault-tolerance, and security considerations will favor the formation of virtual organizations that will develop their own internal commodity markets for computing resources to satisfy the majority of contracts within each organization. Striking a price for options also can be more objective when the utility-pricing strategies discussed in Section 4.7 are considered.

Service request rates and, consequently, the amount of resources needed by service providers obey heavy-tailed distributions: stochastic distributions with large coefficients of variation (the coefficient of variation is a dimensionless number formed
from the ratio of the standard deviation to the mean of a distribution). Overprovisioning, the strategy that makes worst case assumptions about the amount of resources needed, used alone would be very costly and socially wasteful. Last, but not least, the commodities in the subject of this paper are perishable, one cannot store CPU cycles, or communication bandwidth, hence, such resources are either used or lost. Moreover, there is an inherent time lapse associated with any transaction; options could partially hide this interval and provide immediate access to resources when needed. Neither the suppliers, nor the consumers of resources of the system discussed in Section 4.2 know ahead of time the precise amounts of resources they can offer or require. As service providers build their reputation, they will likely experience heavier loads and so need more resources. Service providers should secure excess capacity to cope with the uncertainties pertaining to resource consumption for individual requests, excess load, or other unforeseen events; they should also have the ability, offered by options, to get additional resources, or to sell excess capacity based on the dynamics of the system. In turn, the Market Authority should have the ability to respond to catastrophic events, communication failures, and so on.

In summary, a mix of options and overprovisioning could prove to be an optimal solution for a service-based computing economy; options offer not only the ability to speculate and hedge, but are a critical element of the computer economy.

In Pleiades, resources can be allocated using options, at least until a service provider reaches a steady-state. The following example anticipates the use of a synthetic measure for computing resources, used in Section 4.7, the resource equivalent unit or reu. Assume that a service provider needs 100 reus today. 100 reus can be contracted for
a short period, (one day for example) and call options can be bought for 130 reus with an
expiration date of tomorrow. If the demand for services increases tomorrow and 120 reus
are needed, the option for 120 will be exercised and, in addition, a call options for 140
reus with an expiration date of the following day can bought. If, on the other hand, the
demand for services decreases tomorrow and only 60 reus are needed, the service
provider could exercise the option for 60 reus and buy call options for 70 reus with the
expiration date of the following day. An algorithm based on this idea could minimize
losses due to overprovisioning, subject to the added cost of purchasing options. However,
the price of options is typically only a small fraction of the price of the underlying asset
and the strike price will be close to the current price since the expiration date is soon.

Option pricing in a commodity market for computing resources is certainly an
interesting subject and requires further research. In the next section, we address another
critical issue, pricing the underlying assets in a commodity market for computing
resources.

4.7. A Macroeconomic Model Based on
Price-Utility Functions

In [5], a macroeconomic economic model was introduced that captures traditional
concepts such as consumer satisfaction and provider revenues for resource sharing. This
model enables us to analyze the effect of different pricing strategies on measures of
performance important for consumers and providers alike. Economic models are usually
very complex and require a significant number of parameters and the model discussed in
this section is no exception.
Given a particular set of model parameters, satisfaction reaches an optimum [5]; this value represents the perfect balance between the utility and the price paid for resources. Simulations confirmed that brokers play a very important role and can positively influence the market. From these studies, we learned that consumer satisfaction does not track the consumer utility; these two important consumer performance metrics behave differently under different pricing strategies. Pricing strategies affect the revenues obtained by providers, as well as, the ability to satisfy a larger population of users.

*Pricing Functions.* We consider three pricing functions in Figure 4.7 (a). Given the constant, \( \xi \), the three particular pricing functions we choose are:

1. The price per unit is constant regardless of the amount of resources consumed (linear pricing):
   \[
   p(r) = \xi \cdot r \quad (11)
   \]

2. Discourage consumption: the more resources are used, the higher the average unit price (super-linear pricing):
   \[
   p(r) = \xi \cdot r^d \quad (12)
   \]
   with \( d > 1 \), e.g., \( d = 1.5 \).

3. Encourage consumption: the more resources are used, the lower the average unit price (sub-linear pricing):
   \[
   p(r) = \xi \cdot r^e \quad (13)
   \]
   with \( e > 1 \), e.g., \( e = 0.5 \).
Figure 4.7. (a) Sub-linear, linear, and super-linear price functions. (b) The unit price varies with $\rho$ the load index of the provider.

In Figure 4.7 (b) we analyze the effect of resource abundance; in that case we define the load index $\rho$ as the ratio of total amount of allocated resources to the capacity of the provider and consider three regions: low, medium, and high load index. The pricing strategy for each region is different. We consider two models, EDL - Encourage/Discourage Linear, and EDN - Encourage/Discourage Nonlinear.
**Utility Function**: The utility function should be a non-decreasing function of \( r \), the amount of resources, i.e., we assume that the more resources are allocated to the consumer, the higher the consumer utility. However, when enough resources have been allocated to the consumer, i.e., when some threshold is reached, an increase of allocated resources would bring no improvement of the utility. On the other hand, if the amount of resources is below some threshold the utility is disproportionately low. Thus, we expect the utility to be a concave function that reaches saturation as the consumer gets all the resources it can effectively use. These conditions are reflected by the following equations:

\[
\frac{du(r)}{dr} \geq 0 \quad \text{and} \quad \lim_{r \to \infty} \frac{du(r)}{dr} = 0 \quad (14)
\]

For example, if a parallel application could use at most 100 nodes of a cluster, its utility, reflected by a utility function, will not increase if its allocation increases from 100 to 110 nodes. If we allocate less than 10 nodes then the system may spend most of its time paging and experiencing cache misses and the execution time would be prohibitively high. Different functions can be used to model this behavior and we choose to use a sigmoid:

\[
u(r) = \frac{(r / \omega)^{\zeta}}{1 + (r / \omega)^{\zeta}} \quad (15)
\]

where \( \zeta \geq 2 \) and \( \omega > 0 \) are constants provided by the customer. Clearly \( u(\omega) = 1/2 \) and the utility function is a real number between 0 and 1, \( 0 \leq u(r) \leq 1 \).

A sigmoid is a tilted S-shaped curve, Figure 4.8 (a) that could be used to represent the life-cycles relating to living, as well as many man-made, social, or economical systems. It
has three distinct phases: a starting phase, a maturing phase, and a declining, or aging, phase.

Figure 4.8. (a) A sigmoid is used to model the utility function; a sigmoid includes three phases: the starting phase, the maturing phase, and the aging phase. (b) The satisfaction function for a sigmoid utility function and three linear price functions with low, medium, and high unit price.

Satisfaction Function: A consumer satisfaction function takes into account both the utility provided to the consumer and the price paid for resources. For a given utility $u(r)$, the satisfaction function, $s(u(r), p(r))$, should increase when the price decreases and, for a given price, the satisfaction function should increase when the utility $u(r)$ increases.

These requirements are reflected by the following equations:

$$
\frac{\partial s}{\partial p} \leq 0 \quad \text{and} \quad \frac{\partial s}{\partial u} \geq 0 \quad (16)
$$

Furthermore, a normalized satisfaction function should satisfy the conditions:

- given the price, $p(r)$, the satisfaction function, $s(u(r), p(r))$, should approach the minimum (0) when the utility, $u(r)$, approaches 0 and should approach the maximum (1) when the utility, $u(r)$, approaches infinity:
\( \forall p > 0, \lim_{u \to 0} s(u, p) = 0 \) and \( \lim_{u \to \infty} s(u, p) = 1 \) \hspace{1cm} (17)

- given the utility, \( u(r) \), the satisfaction function, \( s(u(r), p(r)) \), should approach the maximum (1) when the price, \( p(r) \), approaches 0 and should approach the minimum (0) when the price, \( p(r) \), approaches infinity:

\[ \forall u > 0, \lim_{p \to 0} s(u, p) = 1 \text{ and } \lim_{p \to \infty} s(u, p) = 0. \]  \hspace{1cm} (18)

A candidate satisfaction function is:

\[ s(u, p) = 1 - e^{\mu \kappa u p^{-\mu}} \]  \hspace{1cm} (19)

where \( \mu, \kappa \) and \( \varepsilon \) are appropriate positive constants. The satisfaction function based on the utility function in equation (15) is normalized; given a reference price \( \phi \) we consider also a normalized price function and we arrive at a satisfaction function given by:

\[ s(u, p) = 1 - e^{\mu \kappa \phi u (p/\phi)^{-\mu}}. \]

Because \( u \) and \( p \) are functions of \( r \), satisfaction increases as more resources are allocated, reaches an optimum, and then declines, as shown in Figure 4.8 (b). The optimum satisfaction depends on the pricing strategy: not unexpectedly, the higher the unit price, the lower the satisfaction.

The 3D surfaces representing the relationship \( s = s(r, \xi) \) between satisfaction, \( s \), and the unit price, \( \xi \), and the amount of resources, \( r \), for several pricing functions (super-linear, and linear) are presented in Figure 4.9; the surfaces for sub-linear pricing and a cut through the surfaces \( s = s(r, \xi) \) at the constant \( \xi \) are shown in Figure 4.10. As we can see from the cut through the surfaces \( s = s(r, \xi) \), at constant \( \xi \), when we discourage consumption (super-linear pricing) the optimum satisfaction is lower and occurs for
fewer resources; when we encourage consumption (sub-linear pricing) the optimum satisfaction is improved and occurs for a larger amount of resources. These plots reassure us that the satisfaction function has the desired behavior.

![Figure 4.9](image)

**Figure 4.9.** The relationship between satisfaction \( s = s(r, \xi) \) and the unit price \( \xi \) and amount of resources \( r \). The satisfaction function is based on a sigmoid utility function and two price functions: (a) discourage consumption (super-linear); (b) Linear.

*The Role of a Broker:* The role of a broker is to mediate access to resources. In our discussion, we concentrate on optimal resource management policies. A policy is optimal when the satisfaction function, which reflects both the price paid to carry out a task and the utility resulting from the completion of the task, reaches a maximum. A broker attempts to operate at, or near, this optimum. We consider provider-broker-consumer models that involve the set of resource providers, \( R \), the set of consumers, \( U \), and brokers, \( B \). These models assume that a consumer must get all of its resources from a single provider. Brokers have "societal goals" and attempt to maximize the average utility and revenue, in contrast, providers and consumers that have individualistic goals; each
provider wishes to maximize its revenue, while each consumer wishes to maximize its utility for the lowest possible cost.

![Figure 4.10](image)

**Figure 4.10.** The relationship between satisfaction \( s = s(r, \xi) \) between satisfaction \( s \) and the unit price \( \xi \) and amount of resources \( r \). (a) encourage consumption (sub-linear); (b) a cut through the three surfaces at a constant \( \xi \).

To reconcile the requirements of consumers and candidate providers, a broker chooses a subset of providers offering satisfaction above a given threshold; all providers in this subset have an equal chance of being chosen by a given consumer. We call the size of this subset the *satisficing size*, and denote it by \( \kappa \); the word "satisfice" was coined by Nobel Prize winner Herbert Simon in 1957 to describe the desire to achieve a minimal value of a variable instead of its maximum [22]. The resource negotiation protocol consists of the following steps:

- All providers reveal their capacity and pricing parameters to the broker.
- A consumer sends, to the broker, a request with the following information:
  1. the parameters of its utility function;
2. The parameters of its satisfaction function:

3. the number of candidate resource providers to be returned

• The broker executes a brokering algorithm and returns the required list of candidate resource providers to the consumer.

• The consumer selects the first provider from the list and verifies if the provider can allocate the required resources. If it cannot, the consumer moves to the next provider from the list until the resources are allocated by a provider.

• The provider notifies the broker about the resource allocation to the user.

Economic models are notoriously difficult to study. The complexity of the utility, price, and satisfaction-based models precludes analytical studies and in [5] we report on a simulation study. The parameters for our experiments are

• the target utility for consumers, $\tau$

• the satisficing size, $\kappa$; it reflects the extent of the choices given to the consumer by the broker, and

• the demand to capacity ratio, which measures the commitment and, thus, the load placed on providers.

We study the evolution in time of (i) the average hourly revenue, (ii) the request acceptance ratio (the ratio of resource requests granted to the total number of requests), (iii) the average consumer satisfaction, and (iv) the average consumer utility. We investigate the performance of the model for different target utilities, satisficing sizes, and demand to capacity ratios. We study several scenarios, for the linear, EDN, and EDL pricing strategies.
The goal of our simulation study is to validate our choice of utility, price, and satisfaction function, to study the effect of the many parameters that characterize our model, and to gain some intuition regarding the transient and the steady-state behavior of our models. We are primarily interested in qualitative rather than quantitative results, i.e., we are interested in trends, rather than actual numbers.

In our model, the actual shape of the utility function is controlled by the parameters dictated primarily by the application. On the other hand, the satisfaction function mostly reflects the user's constraints. The model inhibits selfish behavior: greedy consumers pay a hefty price and greedy providers who insist on high prices are bypassed. The satisfaction function ensures a balance between the amount of resources consumed and the price paid for them.

The function of a broker is to monitor the system and to tune $\tau$ and $\kappa$ for optimal performance. For example, if the broker perceives that the average consumer utility is too low, it has two choices: increase $\tau$, or increase $\kappa$. At the same time, the system experiences an increase in the average hourly revenue and a decrease of the average consumer satisfaction. The fact that increasing utility could result in lower satisfaction seems counterintuitive, but reflects the consequences of allocating more resources; we increase the total cost possibly beyond the optimum predicated by the satisfaction function. The simulation results shown here are consistent with those when we use linear pricing and simpler models based upon a synthetic quantity to represent a vector of resources.

The EDL pricing strategy leads to the highest average consumer utility and the highest average hourly revenue, while it gives the lowest request acceptance ratio and the
lowest average consumer satisfaction. The EDN pricing strategy allows the highest request acceptance ratio and the highest average consumer satisfaction, while it leads to lower average consumer utility and average hourly revenue than EDL. It is also remarkable that the average consumer satisfaction does not track the average consumer utility. This shows the importance of the satisfaction function. One could argue that in practice it would be rather difficult for users to specify the parameters of their utility and satisfaction function. Of course, this is true in today's environments, but entirely feasible in intelligent environments where such information could be provided by societal services. The advantages of elastic requests for the computational economy of the future are likely to motivate the creation of services that could provide the synthetic parameters required to compute the satisfaction and utility functions.

Even though we limit our analysis to a single broker system, we are confident that the most important conclusions we are able to draw from our model will still be valid for multiple broker systems. These conclusions can be summarized as follows:

- Given a particular set of model parameters the satisfaction reaches an optimum; this value represents the perfect balance between the utility and the price paid for resources.
- The satisfaction does not track the utility,
- Differentiated pricing perform better than linear pricing,
- Brokers can effectively control the computing economy

In an environment with multiple brokers, individual brokers could apply different policies; providers and consumers could join with the one that best matches with their individual goals. The other simplifying assumptions for our analysis, e.g., the uniformity
of the demand to capacity ratio for all resources available at a consumer's site, will most likely have second order effects. The restriction we impose by requiring a consumer to obtain all necessary resources from a single broker is also unlikely to significantly affect our findings.

It is very difficult to make a direct comparison between systems based on different models with different objective functions. The results discussed in [5] are qualitative rather than quantitative; the goal of our work is to show that our formal mathematical model captures and predicts performance trends.

4.8. Conclusions

A service-oriented computing economy based upon resource virtualization has the potential to stimulate the further growth of Web-based service delivery. Here we have described our plans for the Pleiades system that will allow excess computing resources to be bought and sold as in a commodities market. Options enable a service provider in a service-oriented computing economy to minimize its costs while being able to adapt to sudden change of demand for services. Our future work involves developing the implementation of the Pleiades system (please see http://condgraf.ucc.ie/Pleiades/).

Acknowledgments

The research reported in this paper was partially supported by Ernest T. S. Walton award from the Science Foundation of Ireland to Dan C. Marinescu. John P. Morrison gratefully acknowledges the support of the Science Foundation Ireland under the Principal Investigator award scheme. H.J. Siegel was supported by the US National
Science Foundation grant number CNS-061570 and the Colorado State University Abell Endowment.

References


