# Grid Resource Allocation by Means of Option Contracts

Anton Bossenbroek, Alfredo Tirado-Ramos, and Peter M. A. Sloot

***Abstract*—In Grid environments, where virtual organization resources are allocated to users using mechanisms analogue to market economies, strong price fluctuations can have an impact on the nontrivial quality-of-service expected by end users. In this paper, we investigate the effects of the use of option contracts on the quality of service offered by a broker-based Grid resource allocation model. Option contracts offer users the possibility to buy or sell Grid resources in the future for a strike price specified in a contract. By buying, borrowing and selling option contracts using a hedge strategy users can benefit from expected price changes. In this paper, we consider three hedge strategies: the butterfly spread which profits from small changes, the straddle which benefits from large price changes, and the call strategy which benefits from soaring prices. Using our model based on an abstract Grid architecture, we find that the use of hedge strategies augment the ratio of successfully finished jobs to failed jobs. We show that the degree of successfulness from hedge strategies changes when the number of contributed resources changes. By means of a model, we also show that the effects of the butterfly spread is mainly explained by the amount of contributed resources. The dynamics of the two other hedge strategies are best explained by observing the price behavior. We also find that by using hedge strategies the users can increase the probability that a job will finish before the deadline. We conclude that hedging using options is a promising approach to improve resource allocation in environments where resources are allocated by using a commodity market mechanism.**

***Index Terms*—Computational Grids, Grid economies, finance, hedge strategies, options, resource allocation.**

## I. Introduction

DUE TO limitations imposed by distributed computing technologies, high-performance applications often have to be parallelized. In the past the only possibility to execute such applications was by having access to a super computer which could perform the calculations required. This approach poses several issues; for instance, if more capacity is needed, it may not be feasible in the short term or cost effective to build or extend a parallel computer. While personal computers, workstations or parallel computers at the local site may run at full capacity, systems at other sites might be under used [1]. One of the most interesting recent approaches to the effective use and sharing of distributed resources is offered by Grid computing technology.

Manuscript received August 22, 2007; revised June 16, 2008. First published February 6, 2009; current version published February 25, 2009. This work was supported by Dutch Ministry of Education, Culture and Science's Virtual Laboratory for eScience (VL-e) project.

The authors are with the Section Computational Science, Faculty of Science, Universiteit van Amsterdam, 1098 SM Amsterdam, The Netherlands (e-mail: abossenbroek@me.com; a.tiradoramos@uva.nl; p.m.a.sloot@uva.nl).

Digital Object Identifier 10.1109/JSYST.2008.2011255

Grid computing [2] is a distributed computing paradigm that offers new possibilities to design architectures that provide secure and seamless access to multiple distributed data and computational resources. This is achieved by allowing users to share such resources within multiple Grid virtual organizations (VOs), supported by middleware based on open standards.

The vision of Grid Computing is based on the access to computing and data resources from a virtual infrastructure that mimics the electrical industry power Grid. That is, users should not have to care where available computing resources are, as long as there is a reliable and secure way to access them. This persistent and multipurpose infrastructure will eventually come at some cost, which is expected to be justified because it will be amortized over many uses and users [3].

Different categories of Grids are identified in the literature [4]. The most prevailing types of Grids are Computational Grids and Data Grids. Computational Grids aggregate the power of individual computers into the computational capabilities of virtual supercomputers. Data Grids, on the other hand, aim to create aggregated virtual repositories that give users access to vast amounts of data and storage capacity previously unavailable, using transparent higher level services such as replica location services.

In order to provide uniform views and access to resources, Grids aggregate them in a highly controlled fashion [5]. Access is controlled by local policies translated to sharing rules, which clearly define under which conditions the resources can be used. Resource providers and consumers with the same sharing rules form specialized VOs [2]. Examples of such VOs are loosely coupled consortia of institutions and companies with requirements for high computational or storage services, such as cycle providers, storage service providers, or hospitals with large quantities of digitalized image data. Virtual organizations can be similar to a real administrative organization, and may span multiple administrative domains across geographical boundaries.

Many issues have to be addressed to effectively share resources in a Grid, and the issues involved differ per type and purpose of the infrastructure to be built. For instance, to effectively serve as an extension for parallel computers, computational Grids have to provide at least the same quality of service as such systems. Effective sharing of resources makes it possible for users within a virtual organization to perform computations on resources contributed to the VO; the most straightforward way to achieve this, naturally, is to let the user choose which resources to use. Depending on the amount of resources needed, this method might become tedious and inefficient, and a chosen allocation scheme might be not optimal for either the user nor

the Grid. In this respect, resource allocation in Grid environments is a crucial problem.

Regardless of who performs the resource search and selection, be it users or automated systems, decision makers are faced with the difficult task of matching/mapping jobs to resources. Services such as automated resource discovery help to circumvent some of these problems, by publishing information on local resources in a standard manner. Another type of method which can be useful to allocate resources employs a global scheduler to which jobs are submitted to ensure some global optimum. One disadvantage of a global scheduler, as its name implies, is that it is of centralized nature. This may hinder flexibility and scalability in large Grids which consist of dynamic virtual organizations.

Allocation schemes using an artificial economic marketplace may be optimal for both, users and providers under certain conditions. Virtual economic markets would allow users and Grid resource providers to discover information and voluntarily participate in the trade of resources. Mathematical model such as the general equilibrium theory of Walras [6], [7] has to be used to recreate economic markets in a Grid environment [8]. In order to use these theories, we assume a demand function for Grid resources. In our investigation, we only consider implementations of economic markets in Grid environments where the demand for a Grid resource, determined by demand functions, is driven by a monetary value to which we refer as Buyyas (By$).

In addition to a possible optimal allocation scheme, the use of a market mechanism to allocate Grid resources offers more benefits. A market mechanism permits users and resource providers to make their own decision to maximize the quality of service, or profit [9]. Furthermore, markets are intuitive for users [10]. By formulating the allocation problem in economic terms, we can draw upon the vast body of economic research to help understand the behaviour of VOs [11]. Additionally in [12], Wolski *et al.* conclude that using Walras' general equilibrium theorem for economic markets is, although more complicated, more efficient than auctions. Finally, if the currency used in the Grid can be exchanged in a real currency, it would allow to charge users for the use of resources and create an additional incentive for resource providers to share resources within VOs.

### A. Approach

In this paper, we assume that users demand resources by submitting jobs, and that when submitting a job a user specifies how much budget he is willing to pay to process the completion of the job. As resources are assumed to be scarce, it is possible that an excess demand exists at a certain price. The price where this excess demand is zero, or where the aggregated demand for resources matches the aggregated amount of resources shared, is called the spot price[1]. Depending on the behavior of the users and of the resource providers, the spot price of a resource may fluctuate. As remarked by [13], price instability can have negative effects on applications and schedulers which base their decision on the spot price. We argue that strong price fluctuations also make the Grid unreliable in terms of cost of job execution, and therefore unattractive to end users.

[1]In microeconomic literature, this price is referred to as the equilibrium price.

To diminish the negative effects originating from large price fluctuations, we propose the use of *option* contracts. These contracts permit users to take advantage from expected price changes. In contrast to other approaches which seek to reduce the harmful consequences of price fluctuations, options can be implemented as an extension of any Grid economy.

Option contracts are contrast which permit, though not obligates, the holder to buy or sell an underlying asset in the future for a predetermined price. An option which protects a buyer from soaring prices by permitting to buy for a maximum price is called a call option. Sellers can protect themselves by buying a put option contract which permits the holder to sell for a minimum price. The predetermined price at which the underlying asset, in our case a Grid resource, can be bought or sold is referred to as the *strike* price. The future point in time where the right stipulated in the option contract may be exercised is known as the *maturity time*.[2] When an asset is the underlying of an option it is said to be *covered*. By obtaining an option, the option holder transfers the risks to the option issuer, the *writer*. The risk originates from the uncertainty caused by price fluctuations. Since the option writer cannot be exposed to infinite risk a price, or *premium* is charged for the option.

Options can be bought or borrowed and than sold. The buyer of an option is said to take a long position in an option. A short position indicates that a party has borrowed and sold an option. The payoff of a short position is always the inverse of the long position. By combining short and long position in put and call options users can construct *portfolios*. The strategy used to construct a portfolio is referred to as a *hedging strategy*. By using hedging strategies users can benefit from price fluctuations.

In order to use options, we first propose a Grid architecture where resources can be traded as standardized commodities on a Grid resource market place. To implement options we extend the architecture with services which support the use of options. To investigate the potential of options the design is implemented in a simulation. Using the simulation the impact of three different hedging strategy on the perceived quality of service is analyzed. This paper is concluded with a discussion of our conclusions and ideas for future work on the subject.

## II. Related Work

Current research in the use of financial derivatives in computer systems is predominately available in the field of Grid computing [13]–[15], Telecommunication networks [16], and multi-agent environments [17]. In these systems futures and call options are used to minimize risk for risk-averse agents or maximize utility [18]. Two concepts borrowed from microeconomics. To the best of our knowledge no research has yet investigated the possibilities offered by taking long and short positions in options to construct hedging strategies adopted from finance.

The prevalent financial derivatives used in computer architecture are futures. One of the first observations of the use of such contracts to reserve resources on a PDP-1 at Harvard can be found in [19]. In this account, Sutherland observes that this reservation leads to higher utilization compared to other sites.

[2]The options used in this work are commonly known as European options. These options can only be exercised at maturity time.

Load balancing in multi-agent environments using call options is considered in [17]. In their work, Bredin *et al.* conclude that the use of European call options with a zero strike price enables risk-averse agents to reduce the volatility of the completion time.

In [16] the authors consider the use of options to improve the quality of telecommunication. The options used in the latter work are call options which permit the holder to purchase a given quantity at regular time intervals. The author concludes that the use of this style of options permits to reduce the risk of the communication being dropped before service completion.

Different hedge strategies employing long positions in put and call options are researched in [15]. A hedge strategy which is specifically tailored for acquiring a bundle of options with the same maturity time is researched in [14]. This latter strategy is of great interest in Grids where the ability to co-scheduling is desired.

## III. Abstract Architecture

There is no accepted consensus on how to define Grid architectures [20], [21], though architectural characteristics are often defined by the middleware and protocols [5] used to link the system components. The architecture proposed in this paper builds on existing Grid resource management architectures such as [22], [23]. Our architecture consists only of the essential components necessary to investigate the use of options in a Grid environment, where resources are allocated using an economic model and call options can be purchased.

We next discuss the services defined in our Grid architecture. After having identified and defined their functions and roles, we continue with a discussion on their interactions and dynamics.

### A. Services

Each Grid is composed of services which interact to serve the purpose of the Grid architecture. We identify two categories of components: the first category comprises services that manage or facilitate the processing of the jobs submitted by users who can also be represented by agents acting on their behalf. The components in this category are: Local Schedulers, Grid Information Service (GIS), Grid Resource Broker (GRB), and the Job Submission Service (JSS). The second category is made up by services which decide whether resources are available for a job to be processed. In our model, these decisions are primarily based on economic rules. The services in this category are: the Trade Manager, Grid Bank, the Derivative Broker, and the Option Issuing Service.

*1) Local Schedulers:* Local schedulers are gateways which manage access to Grid resources. The Grid resources are modelled as Virtual Workspaces (VWs) [24] running on Virtual Machines (VMs). Using VWs universal resources can be created which support isolation, fain-grain performance control and improved security without any restrictions on software such as operating systems or libraries [25]. As Freeman *et al.* point out [26] this offers the possibility to detach users from resource providers and thereby enable *division of labor* [27].

A disadvantage of using VWs is that VM images can be of considerable size. To circumvent this problem, VM futures have been proposed. These future contracts specify when and for how long a VM can be *leased*. Before the start of the Lease period the VM image is moved to the resource provider site. Once the Lease starts the VM image is run for the specified Lease. At the end of the Lease the VM image is shutdown or suspended and the image is send back to the user.

By issuing a VM future the issuing local scheduler announces when and for how long a VM will be available. We assume that local schedulers never default on a VM futures and that all VMs boast the same features. Furthermore, all the VM futures in the model have the same Lease period and start one Lease period after being issued. Last, it is not possible to obtain access without a VM future. Because of the limitations on this VM future they will be referred to Leases. This Leases are the quantification of VMs, and therefore also Grid resources. By standardizing the access conditions and the underlying resource access methods, Leases can be traded as commodities such as electricity.

*2) Grid Information Services:* Grid Information Service (GIS) stores and makes available the information needed by other services to interact and discover each other. The GIS in this architecture relies on other services to update its database. Consistency of the database compared to the real state of the Grid is fully dependent on the services which commit information to the GIS. Although we are aware that this introduces a single point of failure, this approach is preferred because it greatly simplifies the architecture [28].

*3) Job Submission Service:* The jobs which are submitted by users are bags of tasks. The tasks in the jobs can be processed in parallel and independent of each other. Typically jobs of this class are computationally intensive but have low requirements on memory and data, as commonly found in bioinformatics and molecular biology.

Users submit jobs via a Job Submission Service using a job specification language to describe the requirements of the job. In the proposed architecture, the job submission specification can be used by the JSS to decide whether to acquire specific resources at a specific time. The purchase of resources is central to the architecture, and is done with the current balance of the submitter at the Grid Bank and the remaining budget allowed to be used to process the job. How the demand is computed is not specified in the architecture; the only constraint to the demand function is that it should map a price to a demand for some quantity of Leases. This is necessary in order to use the economic theory described in the introduction.

*4) Resource Brokers:* Resource Brokers assign resources to jobs after having verified that the job can purchase Leases. To reduce communication overhead caused by VM image migration the Resource Broker will attempt to concentrate jobs at the sites of local schedulers. Before a Resource Broker seeks VMs for the job, it verifies at a Grid Bank if the user can pay for the use of the requested Lease. Once a best match is found, the Resource Broker sends the address of the local scheduler to the JSS.

*5) Trade Manager:* The trade manager's sole purpose is to update the GIS with the spot price for a Lease. This spot price is defined as the price at which the aggregated demand for Grid Resources by JSS exactly matches the aggregated contribution by local schedulers. How this price is determined is dependent

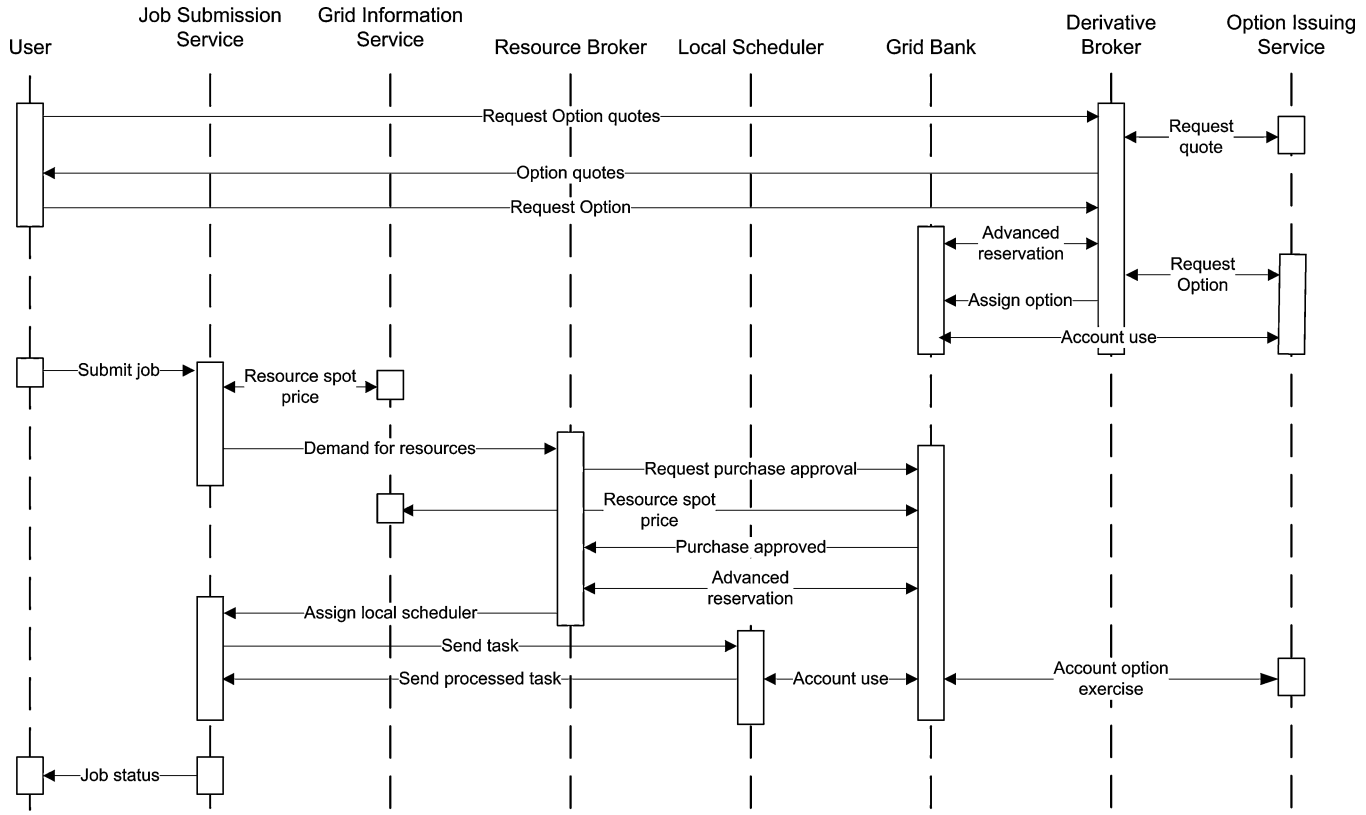



Fig. 1. Interaction between the different Grid services.

on the implementation of the proposed design. For centralized pricing algorithms, such as [29], the Trade Manager may be used to establish a spot price. This functionality is no required in frameworks where the price is established using a peer-to-peer framework, as described in [30]. Although this entity is only present in some economic Grid architectures, it is indispensable in a generic architecture which is to support different pricing mechanisms.

*6) Grid Bank:* Grid bank components are responsible for accounting, verifying and facilitating transactions among Grid entities. Accounting is effectuated when VMs are used and options are sold or borrowed. Verification is required when a VM is assigned to a user and when options are lent. Furthermore, any economic information related to a job or user, including the portfolio, is stored in the Grid Bank. Implementations of Grid Banks in existing Grid projects are discussed in [31]–[33].

*7) Derivative Broker:* A Derivative Broker component consists of services which issue digital financial contracts (derivatives), on behalf of services that provide derivatives which can be used on the Grid. This service is unique to this architecture, as no existing or conceptual Grid architecture in the literature uses derivatives to the best of our knowledge. In our architecture, the Derivative Broker only supports the issuing of options offered by the Option Issuer Service.

Digital contracts representing options are generated by the Derivate Broker and stored at a Grid Bank. The strike price, type (put or call), exercise time, quantity of resources covered by the option and the details of the Option Issuing Service are stored in the contract. By issuing the digital contract the Resource Broker assures that the Option Writer service will be able to fulfill its obligation when the option is exercised.[3]

To obtain a call option contract the future option holder has to enter into a financial transaction with the Derivative Broker. The transaction covers the purchase of an option, and the exact price of an option is determined by the Option Issuing Service.

*8) Option Issuing Service:* The Option Issuing Service determines the price or premium of a call option. For this premium, it offers an option holder to pay out the difference between the strike price of the option and the current spot price of the GR at maturity time, in case this is profitable for the option holder. Options are not directly distributed by the Option Issuing Service, but rather by the Derivative Broker.

As the Derivative Broker will only distribute options of an Option Issuing Service which can fulfill its obligations, the Option Issuing Service should not expose itself to unnecessary risk. This can be prevented by charging a premium for the option. Furthermore, restrictions could be set on the amount of outstanding options or resources covered. Both the computation of a premium and the definition of restrictions are left for implementation.

### *B. Interaction and Dynamics*

The Grid has to process jobs submitted by users. An overview of the interaction between the previously described services is given in Fig. 1.

[3]It is assumed that the market is liquid and that there will always be enough resources to fulfill the contract.

*1) Job Submission and Queueing:* In order to process a job, a user or an agent acting on behalf of the user has to submit it. The job submission specification describes in some standard definition language at least the following information:

- the *Grid Bank account* of the submitter; this information will be needed to approve financial transactions;
- any information on *derivatives* owned by the job submitter;
- the *maximum makespan* before which all the tasks in the job should be processed; the maximum makespan of a job is denoted as $t_{\mathrm{maxspan}}$;
- although not strictly necessary, the specification is preferred to contain also the *maximum budget* which may be used to purchase Grid Resources to process the tasks; from this point forward, the maximum budget specified in the job submission specification is considered as the initial budget of a job, and is denoted as $B$.

The JSS which receives a job description stores it in its job queue. At a specific event the JSS should probe its queue. During the probe at least one of the following three job states is identified:

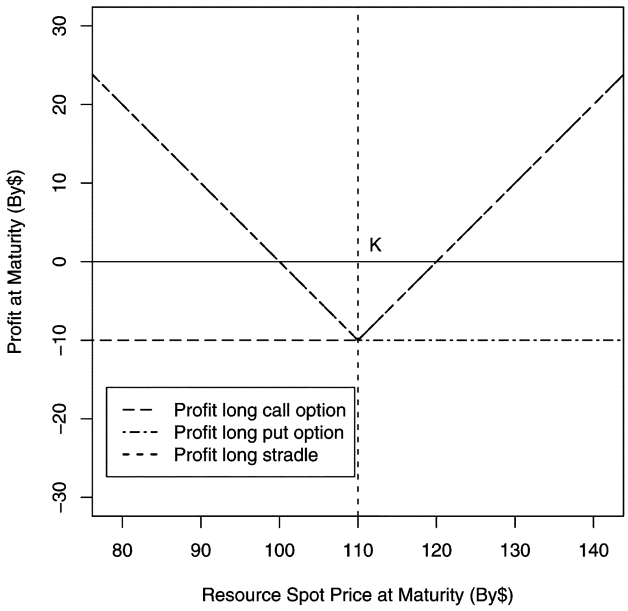
- all the tasks in the jobs are processed. This state of a job is referred to as a successfully finished job;
- the maximum makespan is exceeded. In this case the job is considered to have failed;
- the job is in none of the previously described states.

*2) Price Establishment:* The JSS is in charge of acquiring resources for a determined spot price $p$. This price can be established with any kind of algorithm, but has the property that the Resource market clears at this specific price. Market clearance occurs when the aggregated demand for resources articulated by individual JSS matches the aggregated contribution of resources by Local Schedulers. The lack of excess demand is of great importance, because another price could lead to under- or over utilization of resources. The former causes the Grid to under perform and the latter causes jobs to fail because the number of requests for resources at the Resource Broker will be more than the number of resources available.

The use of a specific pricing scheme to calculate the correct spot price level is left to the implementation of our architecture. Algorithms which could be used are the Walras algorithm [29], the Arrow and Debreu theorem [12] or an implementation of Smale's theorem [34]. The spot price of a Lease will be denoted as $p^*$.

*3) Resource Acquisition:* Once the spot price is known, the JSS submits the request for GRs to the Resource Broker, which inquires the Grid Bank to ensure that the owner can pay for the use of the GR.

A Grid Bank has to approve a request for resource acquisition based on two job properties. The first is that the job owner has sufficient By$ to pay for the use of the GR. After the Resource Broker has obtained an approval from a Grid Bank, it makes a reservation on the account of the user. In case the resource fails before the end of a Lease, the reservation is withdrawn; otherwise the Grid Bank accounts for the use of the resource and transfers the By$ to the account of a Local Scheduler. Advance reservation as described is common practice in credit card companies.




When using a straddle hedge strategy the user takes a long position in a put option and a call option. Both options have the same maturity date and the same strike price. This strategy should be employed when large price fluctuations are expected.

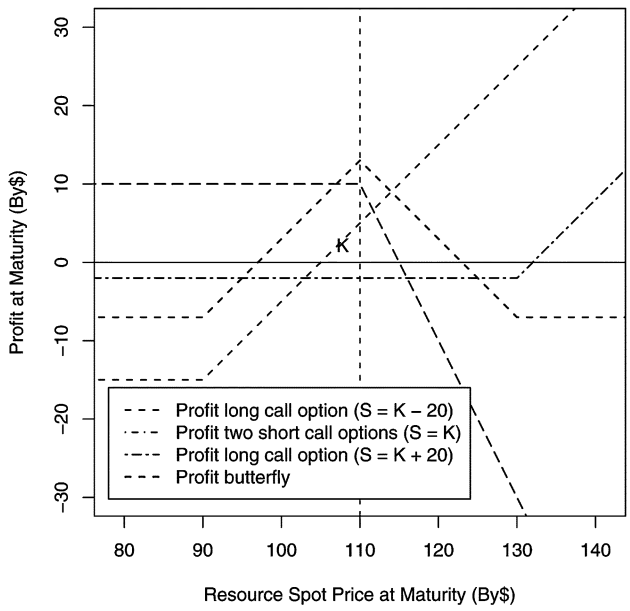
Fig. 2. Payoff of a straddle hedge strategy.

*4) Option Exercising:* At maturity time, an option is exercised when the payoff is positive. As can be seen in Fig. 2, taking a long position in a put option is beneficial when the strike price is higher than the spot price. A call option in contrary is favorable when the strike price is below the spot price. When the holder is in a short position the payoff is the inverse of the long position (see Fig. 3). The payoff of the long position in an option is obtained by exercising the right and immediately resell or buy the Lease at the spot price. The profit of the short position is obtained when the user borrows an option at zero interest from the option writer and sells it to another user through the option writer. In case of a short position, a loss is possible. This is provoked when the option holder exercises the right stipulated in the contract.

All the accounting necessary for the previous process is done by the Option Issuing Service. The costs and benefits are subtracted or added to the current budget. It is left up to the JSS and the job submitter to decide if the payoff should be added to the budget of the job.

*5) Option Issuing:* Before obtaining a digital contract representing an option, the future option holder solicits quotes from the Derivative Broker. The quote holds the premiums, calculated by Option Issuing Services given a strike price $K$, exercise time $T$.

Based on the quote, the future option holder selects a Option Issuing Service. This information is passed to the Derivative Broker which ensures that the Option Issuing Service can fulfill its hedge obligation. Second, the Derivative Issuer requests an advanced reservation for the payment of the option premium or, in case of a short position, the repayment from a Grid Bank.




A portfolio which is used to construct a butterfly spread hedge strategy consists of a long position in two call options and a short position in two call options. All options have the same maturity date but different strike prices. The long position is taken in options with strike price $K - \Delta$ and $K + \Delta$. The short position is taken in two options with strike price $K$. The setup costs are the sum of all the premiums and is always positive. In contrast to a straddle hedge strategy (see Fig. 2) this strategy should be used when no large fluctuations are expected.

Fig. 3. Payoff of a butterfly spread hedge strategy.

The digital contract is then generated and submitted to the Grid Bank, at this point the Grid Bank transfers the premium which was reserved in advance to the account of the Option Issuing Service.

*6) Interest Rate:* In real economies an interest rate exists. As put forward by [35] and later more precisely by [36], an interest rate articulates that an individual prefers present goods to an equal amount of goods in the future. In the architecture this captures the wish of a user to minimize the makespan of a job, and as such, prefers to acquire access to resources at the present time rather than in the future.

To be able to acquire resources and options a user has to deposit By$ at a Grid Bank. Although our architecture uses By$ as a imaginary currency, implementations could use this currency as a virtual currency which can be bought with real currencies. We argue that implementations which use an internal interest rate will thrive. First, because as previously discussed job submitters will be compensated for a slow job execution. Second, the opportunity cost,[4] will be lower than when no internal interest rate would exist.

[4]The opportunity cost, is the cost of the second best alternative.

## IV. Simulation

In order to investigate the potential offered by the use of call options, the abstract architecture is put to test in a discrete time simulation. The simulation emulates the behaviour of the Grid services described in the architecture. To evaluate the performance of the Grid, monitors are implemented at the level of individual services. Traces collected from these monitors are used to compute the performance in terms of basic performance metrics.

We next discuss specific implementation details and parameters used in the Grid services, and then analyze the simulated events. We conclude this section with a discussion about the placement of monitors and the metrics used to measure the performance of the Grid.

### A. Grid Services and Parameters

In the simulation, the JSS receives jobs only from users. When submitting a job, the job consists only of the tasks to compute and a job specification which cites the initial budget, and maximum makespan of a job. The user is modeled to have always sufficient funds stored at the Grid bank.

*1) Job Submission Rate:* Statistics of workload at Grid, VO and region level are analyzed in [37]. Although the authors observe that a two-state Markov modulated Poisson process (MMPP) results in the best approximation of the job arrival rate at a Grid level, we prefer to use the analytically less complex Poisson distribution.

*2) Job Specifications:* Empirical observations by [38] led to the conclusion that process lifetime distributions in Unix systems can be approached with a Bounded Pareto distribution [38]. This has lead to the use of the Bounded Pareto distribution as the distribution of the job size in simulations of a network of heterogeneous computers [39] and Grids [40], and as a distribution of memory demand of jobs in a simulation of distributed systems [41]. Since this has been deduced from empirical evidence and used in models of distributed computing environments, we argue that it is reasonable to assume that the number of tasks in a job follows a Bounded Pareto distribution.

The Bounded Pareto distribution has three parameters: $\alpha$ the exponent of the power law, $k$ the smallest observation, and $l$ the largest observation. The probability density function $\mathrm{BPAR}(k, l, \alpha)$ of the Bounded Pareto distribution is defined as

$$f_p(x) = \frac{\alpha k^\alpha}{1 - (k/l)^\alpha} x^{-\alpha-1} \quad k \le x \le l. \tag{1}$$

The Bounded Pareto distribution has a heavy-tailed property. This property implies that, in the case of job size, most job submissions will consist of jobs with a small amount of tasks.

In the simulation, tasks are quantified in Leases. This makes the number of tasks in a job analogous to the amount of resources needed to process the job. The initial amount of tasks in a job is therefore $J \sim \mathrm{BPAR}(k_{\mathrm{job-size}}, l_{\mathrm{job-size}}, \alpha_{\mathrm{job-size}})$. The remaining amount of unprocessed tasks in a job is denoted as $n$.

No empirical data was either available on the maximum makespan of jobs or on the budget users are prepared to pay to process the tasks in a job. For this reason, we propose

an approach which we argue would best reflect a real Grid environment.

The price the user is prepared to pay, i.e., the budget for a resource, is estimated to be close to the estimated spot price of a resource at time of the job submission. This assumption is based on the presumption that a user will not be prepared to pay much more than the current estimated price. Moreover, users who do not have enough budget available to pay a price considerably close to the estimated spot price are expected to refrain from submitting jobs.

Future prices are estimated using an autoregressive moving average (ARMA) model [42]. This model is commonly used in time series analysis to predict values. The model consists of two parts, the first is an autoregressive part (AR) and the second part is the moving average (MA). The general form of a ARMA$(p, q)$ model is defined as

$$X_t = \varepsilon_t + c + \sum_{i=1}^{p} \varphi_i X_{t-1} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{2}$$

where $\varepsilon_i$ is an error term with a normal distribution with mean zero and standard deviation $\sigma$, and $\theta_i$ is some parameter in the model. In the simulation a ARMA(1,1) model is used with:

$$\theta_1 = \ln\left(\frac{p^*}{p^*_{t-1}}\right). \tag{3}$$

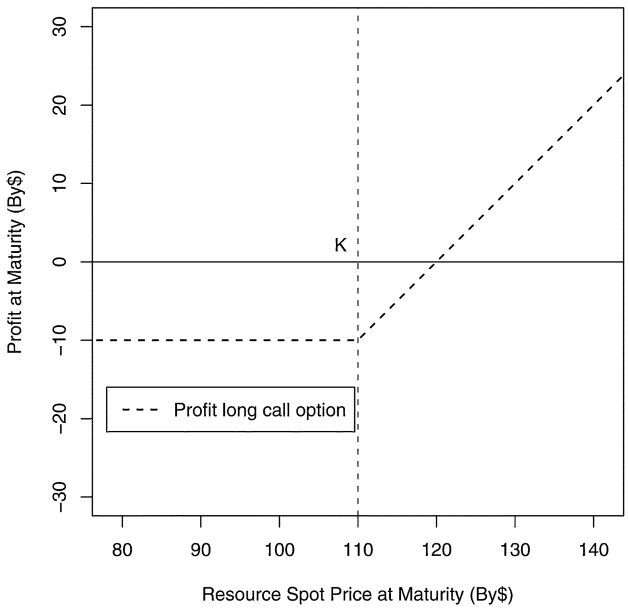
This parameter of the moving average captures the change between the two last prices. The value of $\varphi_1$ explains how the predicted price is dependent on the last price. The model is solved using a least square estimation. This estimation method is highly fit for this application because it assumes an error which follows a normal distribution, which is the same distribution of the error in the ARMA$(p, q)$ model.

The budget per resource is modeled as a random variable from a Gaussian distribution with a mean equal to the estimated price and a standard deviation a fraction $\epsilon$ of the estimated price. The estimated price is computed with the ARMA(1,1) model discussed before. The initial budget $B$ specified in the job specification is equal to $B \sim N(p^e, (\epsilon p^e)^2)$.

The histogram in Fig. 5 visualizes the previous parameters of the job specifications. It shows that a strong concentration of jobs consist of a small amount of tasks and have a normalized budget per resource $BJ/p^e$ close to one.

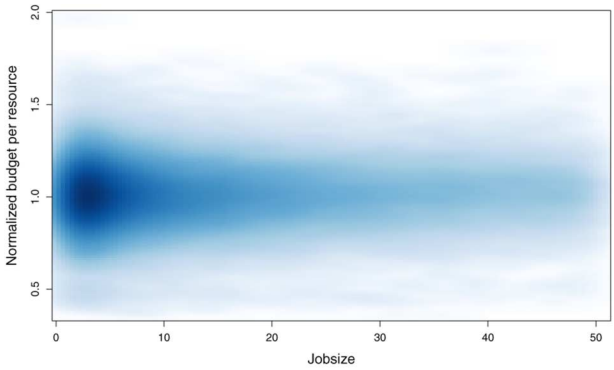
The maximum makespan of a job is assumed to consist of the time needed to process all the tasks sequentially and some slack time. It is presumed that most users will want their job returned quickly whereas just a few will have a more relaxed constrain on the maximum makespan. This leads to the use of the bound Pareto distribution to determine the slack time. As the number of tasks is equivalent to the number of Leases needed for the maximum makespan, $t_{\text{maxpan}}$ is computed as $t_{\text{maxspan}} = J+S$, where $S \sim \text{BPAR}(k_{\text{maxspan}}, l_{\text{maxspan}}, \alpha_{\text{maxspan}})$.

*3) Hedging:* By taking a position in one or more options users can benefit from price fluctuations. The strategy used to decide what position to take in which options with which strike price is called a hedging strategy. In our research, three hedging strategies are used to analyze the potential of options.




A call option hedge strategy uses only a long position in a call option. This strategy is used to benefit from soaring prices..

Fig. 4. Payoff of a call option hedge strategy.




In this histogram the color intensity displays the frequency of a pair occurring. This pair consists of a number of initial tasks in a job, and the budget per resource normalized using the Grid resource spot price at time of submission.

It is clear that most jobs are small and that most budget is concentrated around one. This is the expected behaviour over the probability distributions used to generate the job specifications.

Fig. 5. Histogram of the number of tasks per job.

The first hedging strategy benefits only from soaring prices. The payoff of this hedge strategy is displayed in Fig. 4. As can be seen in this figure, there is only a profit whenever the strike price higher than $K + P$. Considering this payoff this hedging strategy can best be viewed as the strategy necessary for users who wish to protect themselves from soaring prices.

The second hedging strategy, commonly referred to as straddle, benefits from large price fluctuations. To construct the

portfolio for this hedging strategy the user has to take a long position in a put and a call option with the same strike price. The profit of this strategy is shown in Fig. 2. If, at expiry of the options, the spot price is close to the strike price a loss is made. Otherwise a profit is made.

The last strategy uses a long position in two call options and a short position in two other options. This strategy, in financial literature referred to as a call option butterfly spread, permits the user to make profit when the spot price remains close to the strike price. The payoff of the individual options and the total strategy is shown if Fig. 3. Especially in Grid environments where the price is not expected to change much this strategy can be of benefit.

In [43], Nabrzyski *et al.* argue that the users do not want to be derivative traders. We agree with this and therefore the users only specify a hedge rate at submission. This hedge rate is the upper limit of the number of tasks which should be covered. The user is assumed to have sufficient funds to pay the premium of the purchased options and that the user will be able to pay the loss in case of a butterfly spread hedge strategy.

To decrease the makespan of a job, it is more useful to have a higher quantity of options just after the job submission than later in the job's life time. Especially with low hedge rates this is even more important. Based on these constraints the quantity of hedges made with an expiry date $T$ is given by

$$q_T = 2hJN(T, h^2) \tag{4}$$

where $h$ is the hedge rate and $N(T, h^2)$ is the probability density function of a normal distribution with a standard deviation $h$. Since $\int_{-\infty}^{\infty} N(T, h^2)dT = 1$, $\sum_{t=0}^{t_{\text{maxpsan}}} q_t < hJ$.

*4) Resource Demand to Process Tasks:* To process the tasks in a job, a JSS acquires resources based on a demand function. The JSS implementation employs a demand functions which consider the remaining budget which can be used to process the tasks, the number of remaining tasks and the current spot price. Two demand functions are put forward and tested in the experiments.

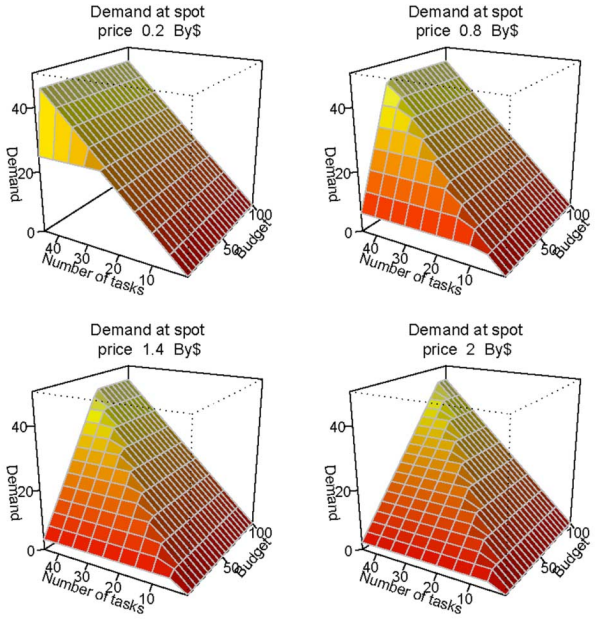
Essential to this model is that any call option associated with the job and with maturity time equal to the current time, should be used to take advantage of high spot prices. Because the Option Issuing Service will pay the option holder before the Grid Bank approves resource purchases, the JSS can use the payoff of the option with an exercise time equal to the current time to acquire resources. Therefore, the JSS uses the budget constrain $\mathcal{B} = b + q_T \max\{0, S - K\}$, where $b$ is the budget remaining and the second part of the equation is the payoff of the option.

The demand function which is used as the baseline demand function only considers the budget of a job and the spot price

$$d_p(\mathcal{B}, p) = \frac{\mathcal{B}}{p}. \tag{5}$$

In Fig. 5, the demand is shown at different spot prices.

A major drawback of the demand function $d_p(\star)$ is that it always consumes the entire budget at once. Consider the case when a job consists of $N = 50$ tasks and the budget is $b = 100$. Using (5), all the tasks can only be computed when the Grid Resource spot price is lower than 2 (see Fig. 5). If this is not the case, all the budget will be used and still some unprocessed tasks will remain in the job. To prevent such job failures a second demand function is needed.




The demand function $d_p(\star)$ leads to lower demand when the price is high. However the budget is always entirely consumed, a feature which is undesirable when $p > B/n$, therefore this function is used as a baseline function.

Fig. 6. Demand function $d_p(\star)$ at different spot prices.

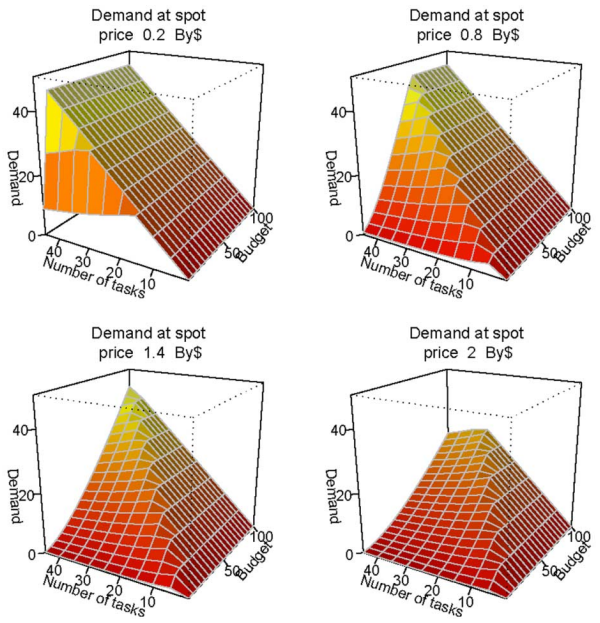
This second demand function considers not only the budget of the job but also the amount of tasks left. It uses these two properties to consider the ratio of the needed budget to the available budget. This factor is used to reduce the demand. The function is defined as

$$d_{\text{bn}}(\mathcal{B}, n, p) = \frac{\mathcal{B}}{(1 + np/\mathcal{B})p}. \tag{6}$$

As can be seen in Fig. 6, the demand of a job decreases as the ratio between the unprocessed tasks and the budget increases. Comparing Fig. 6 with Fig. 5 illustrates how the demand decreases considerably at higher spot prices and low budget. This behavior is used to circumvent the problem discussed previously.

It is important to note that both functions $d_p(\star)$ and $d_{\text{bn}}(\star)$ have a constant return to scale, i.e., the demand does not change when the price and budget increase with the same amount. Furthermore, since $\lim_{p\to 0} d_p(\mathcal{B}, p) > n$ and $\lim_{p\to 0} d_{\text{bn}}(\mathcal{B}, n, p) > n$ both functions are bounded from above, i.e., $d_{\#}(\star) = \max\{n, d_{\#}(\star)\}$.

*5) Contributed Grid Resources:* For tasks to be processed, Local Schedulers have to contribute resources to the Grid. In VOs, the number of resources contributed by a local scheduler depends on sharing rules set up by resource owners. In our simulation, these sharing rules are modeled as policies which describe how many resources are shared at a given price. It is considered that, on the short term adaptation to price changes are




The demand function $d_{bn}(\star)$ leads to a lower demand when the available budget is low, and many tasks still remain in the job. This effect is amplified by a higher price. Compared to $d_p(\star)$ for equal $n$, $p$ and $B$ $d_p(\star) > d_{bn}(\star)$

Fig. 7. Demand function $d_{\mathrm{bn}}(\star)$ at different spot prices.

hard to realize. That is, extending the amount of contributed resource could involve the purchase of the new hardware, and so forth. Furthermore, it is considered that a price increase creates an incentive for resource owners to contribute more resources.

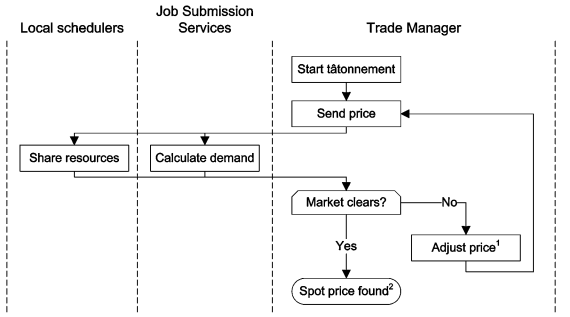
Grids are prone to failure. Therefore, the resource providers are modeled to have a dynamic availability of resources. Theoretical and empirical research on resource availability has been done in the context of optimal checkpointing [44] and fitting probability distributions on availability [45], [46]. In [45], Nurmi *et al.* conclude that both the hyperexponential and the Weibull distribution are usable distributions to model the resource availability in distributed environments where workstations are used as computing nodes. Data on failures of high-performance-computing systems is analyzed in [46], where Schroeder and Gibson conclude that the mean time between failures (MTBF) and mean time to repair (MTTR) are best modeled with the Weibull distribution. Based on the conclusions from both papers, the MTBF and MTTR are therefore simulated to follow a Weibull distribution.

The two-parameter Weibull distribution $\mathrm{WEI}(k, \lambda)$ has a probability density function given by

$$f_w(x) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/k)^k} \tag{7}$$

where $k > 0$ is called the *shape* parameter and $\lambda > 0$ the *scale* parameter.

The model for the number of contributed resources consists of two parts. The first part describes the short term reaction of a price change on the amount of resources contributed to the Grid, and the second models the resource availability.




[1] The price is adjusted using the equation $p'_t = p_t + p_t(D(p_t) - C(p_t))/C(p_t)$, where $p_t$ is the last price, $D(p_t)$ the aggregated demand for resources and $C(p_t)$ the aggregated contribution of resources.

[2] The spot price is the price used in the model.

Fig. 8. Tâtonnement process.

Resource providers are considered to contribute a number of resources depending on the price in addition to a constant amount. Therefore, the contribution is modeled as

$$c(p) = Rp^{\rho}. \tag{8}$$

The assumption that short term adaptations are hard to realize leads to a choice of a value for $\rho$ such that $0 < E_{c,p} = |(\partial c)/(\partial p)(p)/(c)| < 1$[5] and $R \in \mathbb{R}^+$ and varies depending on the MTBF and the MTTR.

*6) Price:* The allocation mechanism used in the abstract architecture is based on a pricing mechanism. This mechanism ensures that the demand for contributed resources matches the amount of contributed Grid resources.

In order to establish a price for a GR, a virtual Grid resource marketplace is created which is organized by the Trade Manager. To establish a price the Trade Manager organizes the market by acting as an auctioneer which matches the demand for Grid resource to the amount contributed. By following an *tâtonnement* process the spot price at which the market clears is found, see Fig. 8. This process is the core of the Walras algorithm [6], which is based on Walras' general equilibrium theory. The algorithm was first used in a distributed environment in [29] and later applied in Grids in [48]. As opposed to the Grid environment discussed in the latter paper, our simulation deals only with the allocation of a single type of resource. Therefore, a simplified version of the algorithm can be used.

In the first step of the tâtonnement process, see Fig. 8, the Trade Manager announces an initial price. Each JSS which wants to acquire Grid resources replies with an aggregated demand of all the jobs scheduled at the JSS. The Local Schedulers reply with the amount of resources which they are prepared to contribute. At this point it is possible that the demand does not match the number of contributed resources.

To find the price at which the market clears, the Trade Manager has to adjust the price to reduce the excess demand to zero. The aggregated demand of all the JSS on the marketplace at a

[5]This choice for the function $c(p)$ is based on the economic theory on price elasticity of supply. This is a key concept in economy, and is well described in economic textbooks such as [47].

price $p$ is denoted as $D(p)$. The aggregated amount of Grid resources contributed by all the local schedulers on the marketplace at a price $p$ is denoted as $C(p)$. The excess demand is then defined as

$$E(p) = D(p) - C(p). \tag{9}$$

To clear the Grid Resource market a price has to be found where $E(p) = 0$. To find the spot price the Trade Manager adjust the price as

$$p' = p + p\frac{E(p)}{C(p)}. \tag{10}$$

This is repeated until a stable price is found. This is price is then the spot price of a GR.

A major advantage of this algorithm is that it recovers quickly from possible communication failures between the participants and the Trade Manager: if during the tâtonnement process some information is lost it will quickly recover from this in the next round [29] or, in case of price stability, use the price from the last round.

*7) Option Pricing:* The Derivative Broker will only issue a derivative from services which will be able to fulfill their obligations. That is, an option issuing service should be able to purchase the number of resources at the spot price, and sell them at the strike price. By issuing an call option contract, the issuer is prepared to take over the risk of the buyer. In order to be able to fulfill its obligation, the issuer is only prepared to do so if it receives a fair premium for the option contract [49].

Grid Resources cannot be stored for later use. This property makes the pricing of options in our model complex, as currently available option pricing techniques assume that the underlying asset of a call option can be owned or borrowed at any time during the lifetime of the option. But, as previously discussed, this property is shared with electricity as a tradable commodity. Therefore, we feel that option pricing mechanisms used on the spot market for electricity are also applicable to call options on Grid Resources.

On commodity markets, derivatives on electricity have been used to allow participants on the electricity market (e.g., energy producers) to protect themselves against price fluctuations. Research on pricing techniques to value these derivatives has been conducted in, e.g., pricing of exotic options [50] and swing options [51].

Major advances in the theory of stock option pricing were achieved by [52] and [53]. In their work, the authors developed a model which has become known as the Black–Scholes model, which was awarded a Nobel price in 1997. One of the important assumptions made in this model is that the price of the underlying asset of the option follows a Brownian motion. Moreover, the Black–Scholes model assumes that an option writer can own or borrow the covered asset at any time. This is not possible in the electricity market.[6]

[6]In most research on pricing electricity options this restriction is circumvented by considering the price of future delivery contracts (futures) as the spot price of electricity. This approach for Grid environments is proposed in [29] but left as future work in the simulation.

The spot price of electricity does not follow a Geometric Brownian motion[7] [54], but follows seasonal trends, such as a price increase during the summer months as in California [55], combined with frequent peaks [56]. Such behavior is commonplace in a Grid environment [57]; due to seasonal holidays Grid users could submit less jobs during a specific period of time. Peaks could also occur due to unforeseen reasons, such as the temporarily disruption of a network connection to a cluster, which would cause a massive shortage of Grid resources within one or more VOs.

Although the driving forces behind the price establishment of Grid resources resemble the dynamics of electricity markets, the market structures differ considerably. After the privatization of the electricity market the supply industry was divided into four categories: 1) generation; 2) transmission; 3) distribution; and 4) retail sales [58]. Furthermore, market rules are laid on the market such as caps on the spot price and production planning.

The particular behavior of the spot price of electricity on wholesale markets has lead to, among others, the development of parametric option pricing techniques based on mean-reverting formulas which assume that on the long term the price will converge to a mean value. In [54], Hjalmarsson argues that although the Black–Scholes prices deviate considerably from the prices found using the nonparametric and parametric models proposed in his work, it is still the best available. Moreover the author claims that it would be difficult to find a model which would achieve better results. We therefore consider the use of Black–Scholes option pricing technique to price options in our simulation.

The precise derivation of the Black–Scholes formula is beyond the scope of this paper and is well covered in textbooks [49]. The formula for a call option is given by

$$P = p^*\Phi(d_1) - Ke^{-rT}\Phi(d_2) \tag{11}$$

where

$$d_1 = \frac{\ln(p^*/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}} \tag{12}$$

and

$$d_2 = \frac{\ln(p^*/K) + (r - \sigma^2/2)T}{\sigma\sqrt{T}} \tag{13}$$

where the function $\Phi(x)$ is the cumulative probability distribution for a standard normal distribution, $r$ the interest rate in the Grid and $\sigma$ the volatility of the price of the Grid resource (see Section IV-A8). The other variables are the same as used throughout this work.

*8) Volatility:* The volatility of the spot price of a Grid resource captures the degree of change of the spot price. In our simulation we use the exponentially weighted moving average (EWMA) to estimate the volatility. The EWMA technique is a special case of the generalized regressive conditional heteroscedasticity (GARCH) technique proposed in [59].

[7]A Geometric Brownian motion is a stochastic process which has the form $dS_t = \mu S_t dt + \sigma S_t dW_t$, where the stochastic process $W_t$ is a Brownian motion with the properties $W_{t+n} - W_t \sim N(0, n)$ and $W_t = 0$. This process is used in finance to model the price of a stock.

In the EWMA model, the square of the volatility $\sigma^2$ is calculated with the equation

$$\sigma_t^2 = \chi\sigma_{t-1}^2 + (1-\chi)u_{t-1}^2 \tag{14}$$

with a smoothing factor $\chi$ within the range $[0, 1)$, and where $u_i$ is defined as $\theta_1$ in (3). As can be deduced from (14), past values have a weight which diminishes exponentially. Although the optimal value of $\chi$ can be found using a maximum-likelihood algorithm, in the simulation a constant value is chosen.

*9) Monitors:* Monitors are implemented at the level of the services to measure overall Grid behavior, as well as the performance of the individual services. A monitor at the user level is used to measure overall statistics on the job specifications and the job states.

### B. Events

The simulations emulate a fixed number of Leases, which are classified into four periods. The simulated events are as follows.

- *User Job Submission:* The user submits a number of jobs. All the aspects of the job specification are also included in this event, e.g., budget specification, option purchasing, etc.
- *Start Tâtonnement:* The Trade manager starts the tâtonnement process to reach a stable price.
- *Stable Price:* The JSS submits the tasks to the local schedulers for processing.
- *Accounting:* All the costs made in the previous events are accounted for at the Grid Bank.
- *JSS Queue Processing:* The JSS processes the queues of jobs.

The events occur sequentially each Lease period

The simulation starts with a warm-up period, meant to fill the queues of the JSS. During this period, no price estimation or hedging is performed. The second period is also a warm-up period, but during this period the price estimation method to compute the budget is used, and hedging is performed.

The third period is the first period where the monitors in the simulation are active and where the obtained measurements are used as results. The last period is a cool-down period. This period is necessary to be certain that every simulation where the number of simulated Lease periods and the exponent of the Poisson distribution for job arrival rate are constant, result in the same amount of jobs used in the measurements. Only jobs submitted during the third period are monitored, the other jobs are ignored.

### C. Metrics

Because Grids are dynamic in their nature, it is difficult to benchmark and evaluate them. Moreover, there is no general consensus on which metrics to use [60], [61]. Since our work concerns the perceived quality of the Grid in terms of job completion, processing time, and costs, we introduce three metrics. The first expresses the effectiveness of the Grid, whereas the other two express the performance of the Grid which in this context does not refer to the computing performance of the Grid, but rather to the amount of time needed and the budget consumed to accomplish a job.

The effectiveness of the Grid is measured using the return states of the processed jobs. The success rate $\varsigma_h^{\{\text{call,btf,str}\}}$ reflects the ratio of the total number of jobs received during one or more Leases to the number of jobs entirely processed before the deadline. The jobs are grouped by hedge strategy and hedge rate $h$. For convenience, the hedge strategies are coded as call, btf (butterfly), and str (stradle hedge). The rate employs the amount of jobs which finished successfully $\mathcal{S}_h^{\{\text{call,btf,str}\}}$ and which failed to be processed $\mathcal{F}_h^{\{\text{call,btf,str}\}}$. The ratio then is defined as

$$\varsigma_h^{\{\text{call,btf,str}\}} = \frac{\mathcal{S}_h^{\{\text{call,btf,str}\}}}{\mathcal{F}_h^{\{\text{call,btf,str}\}} + \mathcal{S}_h^{\{\text{call,btf,str}\}}}. \tag{15}$$

This metric reflects how successful the allocation of Grid resources to jobs is, in terms of job completion. In a production-level Grid the value will ideally tend to converge to one, in contrast to a Grid where many jobs fail to be processed such as an experimental Grid where the success rate will tend to converge to zero.

The presented metrics is used in the next section to evaluate the influence of options on the effectiveness and performance of the Grid.

## V. Results

Grids, and VOs consequently, are dynamic in their nature. Therefore, for options to be fruitful, the increase of the success rate should be scalable. Furthermore, we are interested in the consequences of options on the ancillary quality-of-service. In order to build models which permit to investigate these questions, we generate four datasets by collecting traces from the simulation. The first two datasets consist of the mean success rate of users when varying the fixed amount of shared resources. The second two datasets consist of job information at a certain amount of shared resources.

Section V-A briefly discusses the settings of the essential parameters. We then proceed with a detailed analysis of the scalability of the improvement offered by options. Section V-D discusses the effects of the use of options on the budget and make span of a job.

### A. Parameter Configuration

We set the number of jobs submissions to average 10 000 per Lease, in order to reflect a large sized VO in terms of the number of submissions. As we previously mentioned, options can be implemented as an extension to any resource allocation system which establishes a price for a resource. In such environments, we deem it unrealistic that every user will hedge. Therefore, only 10% of all the jobs are hedged. The jobs are hedged using one of the proposed hedging strategies with a hedge rate ranging from 0.1 to 1 with step size 0.1. This results in an average of 33.3 job submission per Lease per hedge rate for each hedge strategy. Furthermore, resource providers are assumed to react only marginally to price increases. Therefore, the parameter $\rho$ in (8) is set to 0.1. For a detailed account on the implementation of the simulation and the parameter configuration, we refer the reader

TABLE I
ESTIMATIONS OF SUCCESS COUNTS

| Response Variable | | Explanatory Variable | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Success count | | Intercept | | $R$ | | $\bar{\sigma}_t$ | | Estimated | Deviance |
| *dem.* | *strat.* | Estimate | p-value | edf[a] | p-value | edf[a] | p-value | rank | explained |
| $d_{bn}(\star)$ | call | 7.69 | <2e-16 | 6.74 | 2.05e-4 | | | 9 | 72.5% |
| | | 7.69 | <2e-16 | | | 8.22 | 8.56e-8 | 9 | 89.3% |
| | btf | 7.69 | 5.26e-11 | 7.86 | 2.33e-4 | | | 9 | 74.1% |
| | | 7.69 | 2.3e-10 | | | 1 | 1.29e-4 | 2 | 42.4% |
| | str | 7.90 | <2e-16 | 5.49 | 1.07e-4 | | | 9 | 64.8% |
| | | 7.90 | <2e-16 | | | 7.37 | 1.01e-5 | 9 | 81% |
| $d_p(\star)$ | call | 5.38 | <2e-16 | 5.89 | 3.48e-4 | | | 9 | 74.2% |
| | | 5.38 | <2e-16 | | | 8.63 | 1.67e-5 | 9 | 86.3% |
| | btf | 3.50 | 3.03e-8 | 7.18 | 2.86e-8 | | | 9 | 92.5% |
| | | 3.50 | 3.03e-8 | | | 8.60 | 1.84e-5 | 9 | 86.1% |
| | str | 5.50 | <2e-16 | 6.08 | 4.34e-4 | | | 9 | 73.9% |
| | | 5.50 | <2e-16 | | | 8.50 | 3.97e-4 | 9 | 78.8% |

This table summarizes the information on the parameters of the GAMs found. From the table can be concluded that for the success count of the straddle and call hedge strategies can best be explained with the volatility. The butterfly spread instead can best be explained with the number of contributed resources.
[a] edf: estimated degrees of freedom

to the source code of the simulation.[8] The random number used in the simulation are generated using the Mersenne Twister generator [62] part of the GNU Scientific Library [63].

### B. Scalability of the Success Rate

Two data sets are constructed by altering the number of contributed resources [the parameter $R$ in (8)] for the two demand functions. For each value of $R$ traces from 500 Leases are collected after a warm up period of 200 Leases. At the end of each Lease the return status of the jobs returned during that Lease with the same hedge rate and strategy are aggregated, in order to compute the success rate [see (15)]. At the end of the simulation, samples are constructed for each hedge rate and strategy. The baseline is the situation where no hedging is used. This is comparable to the success rate which would be achieved when allocating resources using a commodity market without options such as [12].

To analyze if the success rate at $h > 0$ is larger than the success rate at $h = 0$, the mean of the samples for each hedge rate and hedge strategy are compared with the baseline. This is possible since the errors in the samples are normally distributed. Using a t-test we decide if the means are significantly different at a 5% level. The number of samples which fulfill both constrains are counted by hedge strategy. From this point forward these counts will be referred to as the success counts.

Based on a number of simulations, we observed that the success counts of all the hedge strategies demonstrate local nonlinearity relations with either the number of contributed resources or the volatility. On this basis, we reject the use of (Generalized) Linear Models (GLM). Studies observing data with similar patterns have successfully applied Generalized Additive Models (GAM)[64], [65]. To construct our model, we use GAMs [66] with thin plate splines [67]. The local nonlinearity already demonstrates that the number of contributed resources has effects on the success counts. To analyze if this effect is significant at a 5% level we construct a model where the success count is the response variable and the number of contributed resources is the explanatory variable. The essential information from the GAM is displayed in Table I.

However for the success counts when using a straddle or call option hedging strategy, the number of contributed resources is not the best explanatory variable. The volatility $\sigma$ of the price (see (14)) results in a better estimation of the success count in these cases. Based on the intercept and the p-value of the intercepts we can conclude that hedge strategies are more beneficial when using $d_{\mathrm{bn}}(\star)$. We attribute this to the fact that the standard deviation of the volatility in a VO which uses$d_p(\star)$ is a factor 1.4 larger (0.57 to 1.36). A second explanation can be found in the models which can be used to explain the volatility based on the number of contributed resources.

The data on the volatility exhibits a parametric relation between the volatility and the number of contributed resources. When using $d_p(\star)$ the volatility has to be transformed by taking the $\log()$. This is necessary to have an error factor which is normally distributed. Using this data, we find the model

$$\log(\bar{\sigma}_t) = 2.41 - 5.57 \cdot 10^{-4} R.$$

Combining this model with our previous conclusion, we claim that when few resources are available the low success counts can be improved by increasing the number of resources. However, this is only the case when the number of contributed resources remains under the quantity of resources necessary for a success rate of 1.

[8]The source code can be retrieved using svn. See for more details http://code.google.com/p/cgsim/. The code is available under the terms of the GPL v3.

The model for the volatility when using $d_{\mathrm{bn}}(\star)$ defers from the previous model. Here the error is normally distributed which simplifies our analysis. The best parametric explanatory model for the volatility is

$$\bar{\sigma}_t = 1.15 - 3.25 \cdot 10^{-8} R^2.$$

This model shows us that when the success count is low due to a high volatility the VO has to be expanded or reduced in terms of the number of contributed resources to increase the success count. Similar with our conclusion for the $d_p(\star)$ case, this only holds as long as the number of contributed resources remains below the case where the success rate is 1.

The intercepts of the GAM show that when using $d_{\mathrm{bn}}(\star)$ there is no large difference between the straddle and the call option hedge strategies. This is confirmed by a Wilcoxon rank sum test. The test gives a p-value of 0.72 when comparing the success count of the straddle with the call strategy. Comparing the straddle and the call with the butterfly returns 0.03 and 0.08, respectively. Together with the mean which is 7.90, 7.69, and 5.51, respectively, we can conclude that the butterfly spread hedge strategy is inferior to the straddle hedge strategy. We can make the same observations in a VO where the JSS uses the demand function $d_p(\star)$. However, the success counts of the butterfly spread are not significantly different at a 5% level for the different demand functions.

### C. Impact of the Hedge Strategy on the Budget and Make Span

The use of hedge strategies permits to increase the amount of jobs which tasks are entirely processed before the deadline. However we are also interested in the effects of options on the auxiliary quality of service: the budget necessary to process a job and the make span of a job. Furthermore, we are interested if the effects of hedging is uniform for different job sizes.

*1) $d_p(\star)$ Case:* We analyze a data set consisting of 85 jobs for every hedge rate and hedge strategy using the demand function $d_p(\star)$ at $R = 6700$. The latter number is chosen because at this number all the hedge strategies have a high success count. The jobs are selected at random from a larger sample to minimize possible covariance.

To explain the return status of a job a model is build

$$\mathrm{logit}(p) = -0.90 + 2.62 \log\left(\frac{BJ}{p^e}\right) + 2.87h \qquad (16)$$

where $p$ is the probability of success. There is no difference in the p-values of the parameters of the model which would indicate if the hedge rate is a better explanatory variable than the normalized budget per resource (NBR) $BJ/p^e$. Only by building a model where only one of the two variables is chosen as an explanatory variable we find that the Akaike's Information Criterion for the model which uses the NBR is lower than for the model which uses the hedge rate. It is important to note that the job size is not a significant explanatory variable for the return status.

The model found indicates that the hedge rate can be used to compensate a small NBR. To demonstrate this we select only the jobs which have a $\log(\mathrm{NBR})$ below the first quantile $(-0.25)$. Using the same explanatory variable as in (16), we find a new model where $H$ is the only significant explanatory variable. The model is defined as

$$\mathrm{logit}(p) = -1.49 + 3.01h. \qquad (17)$$

From this model, it is clear that the probability for a job to return finished is lower than for jobs with a higher NBR. However, by hedging the user can improve the probability.

The used budget, which is defined as the remaining budget minus the initial budget and the portfolio setup cost, is clearly influenced by the hedge rate for the job which finish successfully. In a linear model consisting of the hedge rate, the job size, and the hedge strategy, the hedge rate has a coefficient of $-52.84$ compared to $-2.57$ for the job size. Moreover, the straddle hedge strategy has a coefficient $-59.77$. On these grounds, we conclude that the straddle hedge strategy is very costly and that the primary costs are caused by the hedge strategy (the model has p-value $\approx 0$, $R^2 = 0.12$, the low $R^2$ value is caused by large outliers).

Due to the definition of the demand function $d_p(\star)$ the make span is always one Lease when not hedging. The hedging strategies can in this case not be used to reduce the make span.

*2) $d_{\mathrm{bn}}(\star)$ Case:* We create a sample comparable to the previous data set in order to analyze the effects of options in a VO where the demand function $d_{\mathrm{bn}}(\star)$ is used by the JSS.

In this case, the model to explain the probability for success is

$$\mathrm{logit}(p) = -0.90 + 2.50 \log\left(\frac{BJ}{p^e}\right) + 2.87h.$$

The parameters of the model are only marginally different from the ones in (16). This indicates that although the environment differs the NBR and hedge rate have the same influence on the final status. This shows that our approach is indifferent to the demand function.

However, this is not the case for the more specific case where jobs have a NBR below the first quantile (in this case $-1.09$). For these jobs the explanatory model is given by

$$\mathrm{logit}(p) = -0.54 + 1.44h$$

these values are less impressive than the parameters in model (17). When using the demand function $d_{\mathrm{bn}}(\star)$, jobs with a low NBR benefit less from hedging than in an environment where the function $d_p(\star)$ is used.

In contrast to the previous case, there is no statistical evidence which shows that the remaining budget is dependent on the hedge rate. Although it is dependent on the straddle hedge strategy (a coefficient of $-1192$ in a model with a p-value $\approx 0$ and $R^2 = 0.26$, as with the model for $d_p(\star)$ the low $R^2$ is caused by large outliers). To error of the time left ratio is exponentially distributed. Therefore, we transform it by taking the log of the ratio. The best model $(\mathrm{p{-}value} \approx 0, R^2 = 0.65)$ of the time consumed by successfully finished jobs is explained by

$$\log\left(\frac{t - t_{\mathrm{start}}}{J}\right) = -0.47 - 0.11 \log\left(\frac{BJ}{p^e}\right) - 0.09J.$$

This demonstrates that the hedge rate has no influence on the time consumed ratio.

### D. Summary

The success count of the call and straddle hedge strategy are best explained by the degree of price volatility, which has a log-linear relation with the number of contributed resources in case of $d_p(\star)$, and a quadratic relation in case of $d_{\mathrm{bn}}(\star)$. The butterfly spread, however, is best explained by the amount of contributed resources. All the hedge strategies have a lower success count when using $d_p(\star)$. By analyzing two data sets where the success counts are high, we show that the probability for a job to finish successfully is most sensitive to the hedge rate. When considering jobs which are submitted with an NBR below the first quantile the hedge rate is the only significant explanatory variable. We argue that this demonstrates that hedging is favorable for users who do not have enough funds to pay the current spot price. Lastly we found that the time left ratio can be increased more by increasing the hedge rate than by increasing the NBR. Finally, we have not found evidence that one hedge strategy is significantly better than the other two.

## VI. Conclusion and Future Work

In this paper, we proposed an abstract architecture for a computational Grid model, where Grid Resources are allocated to jobs using a trading mechanism analogous to a market economy. At the core of our architecture's scheduling system lies a Grid Resource market. On this market, the demand for and the contribution of Grid Resources are aggregated, and the price where both sums are equal is called the spot price. This spot price fluctuates constantly due to dynamic state of Grids.

We argue that fluctuations in the spot price can degrade the nontrivial quality of service offered by Grids. To counteract this deterioration, and even profit from price fluctuations, we propose the use of option contracts on Grid resources. These contracts permit to buy or sell a resource at a predetermined price in the future. By buying or borrowing and selling a combination of contracts, the user can setup hedge portfolios. These portfolios are constructed to benefit from expected price changes. We find that the butterfly spread is beneficial when there are only small price changes, whereas the long straddle is of use when users expect large price fluctuations. The call option strategy protects users against soaring prices.

Unique to the proposed model are two particular Grid services. The first service is a derivative broker. This service issues derivatives on behalf of the second service which is the option issuing service. The purpose of the derivative broker is that it ensures that the option service will be able to fulfill its obligations specified in the option contract. We propose these services as an extension to existing architectures where resources are traded as commodities.

We claim that hedging improves the quality-of-service in a Grid by causing more jobs to be processed before the deadline. Moreover, we show that by increasing the number of options in a portfolio a user can increase the probability of a job finishing before the deadline more than by increasing the budget available at submission. This effect is strongest for jobs which have a initial budget per resource below the first quantile. The cost of running a job are only significantly increased when using a straddle to hedge. However, this strategy is the most successful one.

By changing the number of contributed resources we show that the improvement offered by the call and straddle strategies are more dependent on the price behavior than on the number of contributed Grid resources. For the butterfly spread however, the number of contributed resources is the predominant reason.

We believe that interesting issues in future research include more accurate pricing systems of the options by observation of the price behavior. Since it is not possible to store Grid Resources, a study on the use of future—or forward contracts could consider these contracts to be analogous to the ownership of Grid Resources. This would pave the way to risk reduction for option issuing services.

## Acknowledgment

The authors would like to thank D. Kandhai, D. van Albada, and R. Buyya for their helpful comments and suggestions.

## References

[1] I. Foster and C. Kesselman, "Computational grids," *CERN European Organization for Nuclear Research—Reports—CERN*, vol. 8, pp. 87–114, 1998.

[2] I. Foster *et al., The Grid 2: Blueprint for a New Computing Infrastructure*. San Francisco, CA: Morgan Kaufmann, 2003.

[3] I. Foster and A. lamnitchi, "On death, taxes, and the convergence of peer-to-peer and grid computing," in *Proc. 2nd Int. Workshop Peer-to-Peer Syst. (IPTPS)*, 2003, pp. 118–128.

[4] M. Bote-Lorenzo, Y. Dimitriadis, and E. Gómez-Sánchez, "Grid characteristics and uses: A grid definition," in *Proc. 1st Euro. Across Grids Conf. (CD)*, Santiago de Compostela, Spain, 2004, vol. 2970, pp. 291–298.

[5] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *Int. J. High Perform. Comput. Appl.*, vol. 15, no. 3, pp. 200–222, 2001.

[6] L. Walras, *Éléments d'économie politique pure; ou, Théorie de la richesse sociale*. Lausanne, France: L. Corbaz & cie, etc., 1874.

[7] K. Arrow and G. Debreu, "Existence of an equilibrium for a competitive economy," *Econometrica*, vol. 22, no. 3, pp. 265–290, 1954.

[8] P. Tucker, "Market mechanisms in a programmed system," *Dept. Comput. Sci. Eng., Univ. California, La Jolla*, vol. 130, 1998.

[9] R. Buyya, D. Abramson, and S. Venugopal, "The grid economy," *Proc. IEEE*, vol. 93, no. 3, pp. 698–714, Mar. 2005.

[10] R. Wolski, J. Brevik, J. S. Plank, and T. Bryan, *Grid Resource Allocation and Control Using Computational Economies*. New York: Wiley, 2003, pp. 747–771.

[11] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, "Economic models for resource management and scheduling in grid computing," *Concurrency Computation: Practice Experience*, vol. 14, pp. 1507–1542, 2002.

[12] R. Wolski, J. S. Plank, T. Bryan, and J. Brevik, "G-commerce: Market formulations controlling resource allocation on the computational grid," 2001.

[13] G. Cheliotis, C. Kenyon, R. Buyya, and A. Melbourne, "Grid Economics: 10 lessons from finance," GRIDS Lab and IBM Research Zurich, Melbourne, Tech. Rep., Apr. 2003, 2003, Joint technical report.

[14] L. Rasmusson, *Evaluating Resource Bundle Derivatives for Multi-agent Negotiation of Resource Allocation*. London, U.K.: Springer-Verlag, 2001, pp. 154–165.

[15] O. Baqueiro, W. Van der Hoek, and P. McBurney, "The performance of option-trading software agents: Initial results," *Lecture Notes in Economics and Mathematical Systems*, vol. 599, p. 113, 2007.

[16] N. Semret and A. Lazar, "Spot and derivative markets in admission control," in *Proc. Int. Teletraffic Congr.*, 1999, pp. 925–941.
[17] J. Bredin, D. Kotz, and D. Rus, "Trading risk in mobile-agent computational markets," Jul. 2000. [Online]. Available: http://www.cs.dartmouth.edu/~dfk/papers/bredin:risk.ps.gz
[18] M. de la Maza, D. Yuret, R. Brooks, and P. Maes, Eds., "A futures market simulation with non-rational participants," *Artificial Life*, vol. IV, pp. 325–330, 1994.
[19] I. Sutherland, "A futures market in computer time," *Commun. ACM*, vol. 11, no. 6, pp. 449–451, 1968.
[20] Z. Nemeth and V. Sunderam, "A formal framework for defining Grid systems," in *Proc. Cluster Comput. Grid 2nd IEEE/ACM Int. Symp. (CCGRID)*, 2002, pp. 188–197.
[21] A. Tirado-Ramos, "Collaboratories on the Grid, collaborative software architectures for interactive biomedical applications," *Thesis*, pp. 113–119, 2007.
[22] R. Buyya, D. Abramson, and J. Giddy, "Nimrod/G: An architecture for a resource management and scheduling system in a global computational Grid," presented at the Proc. 4th Int. Conf. High Perform. Comput. Asia-Pac. Reg. (HPC ASIA), Beijing, China, 2000.
[23] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Tuecke, "A resource management architecture for metacomputing systems," in *Lecture Notes in Computer Science*. New York: Springer, 1998, pp. 62–82.
[24] K. Keahey, I. Foster, T. Freeman, X. Zhang, and D. Galron, "Virtual workspaces in the Grid," *Lecture Notes in Computer Science*, vol. 3648, pp. 421–431, 2005.
[25] R. Figueiredo, P. Dinda, and J. Fortes, "A case for Grid computing on virtual machines," in *Proc. 23rd Int. Conf. Distrib. Comput. Syst.*, 2003, pp. 550–559.
[26] T. Freeman, K. Keahey, I. Foster, A. Rana, B. Sotomoayor, and F. Wuerthwein, "Division of labor: Tools for growing and scaling Grids," in *ICSOC*, 2006, vol. 4294, pp. 40–51.
[27] A. Smith, *An Inquiry Into the Nature and Causes of the Wealth of Nations*. Dublin, Ireland: Messrs. Whitestone & etc., 1776.
[28] A. Naseer and L. Stergioulas, "Resource discovery in Grids and other distributed environments: States of the art," *Multiagent Grid Syst.*, vol. 2, no. 2, pp. 163–182, 2006.
[29] J. Cheng and M. Wellman, "The WALRAS algorithm: A convergent distributed implementation of general equilibrium outcomes," *Computational Economics*, vol. 12, no. 1, pp. 1–24, 1998.
[30] L. Xiao and S. Boyd, "Fast distributed algorithms for optimal redistribution," *J. Opt. Theory Appl.*, 2003.
[31] E. Elmroth, P. Gardfjäll, O. Mulmo, and T. Sandholm, "An OGSA-based bank service for Grid accounting systems," in *Applied Parallel Computing. State-of-the-Art in Scientific Computing. Lecture Notes in Computer Science*. New York: Springer Verlag, 2004.
[32] A. Barmouta and R. Buyya, "GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2003, p. 245a.
[33] R. Buyya, D. Abramson, and J. Giddy, "An economy driven resource management architecture for global computational power Grids," in *Proc. Int. Conf. Parallel Distrib. Process. Techn. Appl. (PDPTA)*, Las Vegas, NV, Jun. 2000, pp. 26–29.
[34] G. Stuer, K. Vanmechelen, and J. Broeckhove, "A commodity market algorithm for pricing substitutable Grid resources," *Future Generation Comput. Syst.*, vol. 23, no. 5, pp. 688–701, 2007.
[35] E. Böhm-Bawerk, *Capital and Interest*. New York: Brentano's, 1922.
[36] I. Fisher, *The Theory of Interest*. New York: Kelley and Millman, 1930, 1954.
[37] H. Li, M. Muskulus, and L. Wolters, *Modeling Job Arrivals in a Data-Intensive Grid*. Berlin, Germany: Springer, 2006.
[38] M. Harchol-Balter and A. Downey, "Exploiting process lifetime distributions for dynamic load balancing," *ACM Trans. Comput. Syst. (TOCS)*, vol. 15, no. 3, pp. 253–285, 1997.
[39] X. Tang and S. Chanson, "Optimizing static job scheduling in a network of heterogeneous computers," in *Proc. Int. Conf. Parallel Process.*, 2000, pp. 373–382.
[40] L. He, S. Jarvis, D. Spooner, X. Chen, and G. Nudd, "Dynamic scheduling of parallel jobs with QoS demands in multiclusters and Grids," in *Proc. 5th IEEE/ACM Int. Workshop Grid Comput. (Grid)*, 2004, pp. 402–409.
[41] X. Zhang, Y. Qu, and L. Xiao, "Improving distributed workload performance by sharing both CPU and memory resources," in *Proc. 20th Int. Conf. Distrib. Comput. Syst.* , 2000, pp. 233–241.
[42] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis, Forecasting and Control*, 4th ed. Hoboken, NJ: Wiley, 1990.
[43] J. Nabrzyski, J. Weglarz, and J. Schopf, *Grid Resource Management: State of the Art and Future Trends*. New York: Kluwer, 2003.
[44] J. Plank and W. Elwasif, "Experimental assessment of workstation failures and their impact on checkpointing systems," in *Proc. 28th Int. Symp. Fault-Tolerant Comput.*, 1998, pp. 48–57.
[45] D. Nurmi, J. Brevik, and R. Wolski, "Modeling Machine Availability in Enterprise and Wide-Area Distributed Computing Environments," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer, 2005.
[46] B. Schroeder and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," in *Proc. Int. Conf. Dependable Syst. Netw. (DSN)*, 2006, pp. 249–258.
[47] K. Case and R. Fair, *Principles of Economics*. Upper Saddle River, NJ: Prentice-Hall, 2002.
[48] K. Subramoniam, M. Maheswaran, and M. Toulouse, "Towards a micro-economic model for resource allocation in Grid computing systems," in *IEEE Canadian Conf. Elect. Comput. Eng. (CCECE)*, 2002, vol. 2, pp. 782–785.
[49] J. Hull, *Options, Futures, and Other Derivatives*. Upper Saddle River, NJ: Prentice-Hall, 2003.
[50] S. Deng, B. Johnson, and A. Sogomonian, "Exotic electricity options and the valuation of electricity generation and transmission assets," *Decision Support Syst.*, vol. 30, no. 3, pp. 383–392, 2001.
[51] J. Keppo, "Pricing of electricity swing options," *J. Derivates*, vol. 11, pt. 3, pp. 26–43, 2004.
[52] F. Black and M. Scholes, "The pricing of options and corporate liabilities," *J. Political Economy*, vol. 81, no. 3, pp. 637–654, 1973.
[53] R. Merton, "Theory of rational option pricing," *Bell J. Econom. Manag. Sci.*, vol. 4, no. 1, pp. 141–183, 1973.
[54] E. Hjalmarsson, "Does the Black-Scholes formula work for electricity markets?: A nonparametric approach," 2003.
[55] R. Brown and J. Koomey, "Electricity use in California: Past trends and present usage patterns," *Energy Policy*, vol. 31, pp. 849–864, Jul. 2003.
[56] C. Jong and R. Huisman, "Option formulas for mean-reverting power prices with spikes," 2002.
[57] C. Kenyon and G. Cheliotis, "Architecture requirements for commercializing Grid resources," in *Proc. 11th IEEE Int. Symp. High Perform. Distrib. Comput. (HPDC-11)* , 2002, pp. 215–224.
[58] F. A. Wolak, *Market Design and Price Behavior in Restructured Electricity Markets: An International Comparison*. New York: Kluwer, 2000, ch. 8, pp. 127–149.
[59] T. Bollerslev, "Generalized autoregressive conditional heterocedasticity," *J. Econometrics*, vol. 31, pp. 307–327, 1986.
[60] Z. Nemeth, G. Gombas, and Z. Balaton, "Performance evaluation on Grids: Directions, issues, and open problems," in *Proc. 12th Euromicro Conf. Parallel, Distrib. Netw.-Based Process.*, 2004, pp. 290–297.
[61] Z. Németh, "Grid performance, Grid benchmarks, grid metrics. Cracow Grid workshop," in *Proc. 3rd Cracow Grid Workshop*, Cracow, Oct. 2003, pp. 34–41.
[62] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simulation*, vol. 8, no. 1, pp. 3–30, 1998.
[63] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi, *GNU Scientific Library Reference Manual—Revised Second Edition (vl.8)*. Bristol, U.K.: Network Theory Ltd., 2006.
[64] T. Hastie and R. Tibshirani, "Generalized additive models: Some applications," *J. Amer. Stat. Assoc.*, vol. 82, no. 398, pp. 371–386, 1987.
[65] N. Beck and S. Jackman, "Beyond linearity by default: Generalized additive models," *Amer. J. Political Sci.*, vol. 42, no. 2, pp. 596–627, 1998.
[66] T. Hastie and R. Tibshirani, "Generalized additive models," *Stat. Sci.*, vol. 1, no. 3, pp. 297–310, 1986.
[67] J. Duchon, "Splines minimizing rotation-invariant semi-norms in Sobolev spaces, ser," *Lecture Notes Constructive Theory of Functions of Several Variables*, pp. 85–100, 1977.

**Anton Bossenbroek** is a second year research master student in Computational Science at the University of Amsterdam, Amsterdam, The Netherlands. He specializes in computational finance. Before his master, he studied computer science and econometrics at the University of Amsterdam. His past research interest was applications of economic and financial theory in computer systems. Currently, his research is focused on applications of numerical methods for stochastic optimal control theory.

**Alfredo Tirado-Ramos** received the Ph.D. degree in computational science from the University of Amsterdam's Informatics Institute, Amsterdam, The Netherlands.

He is currently a researcher with the University of Amsterdam's Informatics Institute. He regularly chairs committees in international conferences on biomedical informatics, intelligent information technology, and teaching computational science. Previously, he worked on high-speed teleradiology at the University of Arizona, Tucson, and at the Philips Research North America, New York, on modeling and methodologies in health informatics interoperability. He has coauthored three patent applications and more than 30 publications in refereed journals, books, and international conferences.

Dr. Tirado-Ramos is currently Assistant Editor-in-Chief of *Future Generation Computer Systems, The International Journal of Grid Computing*.

**Peter M. A. Sloot** studied chemistry and physics and conducted his BioComputing Ph.D. work at the Dutch Cancer institute (NKI), Amsterdam, The Netherlands, with Prof. C. Figdor.

In 1996, he was awarded a distinguished endowed Professor position in numerical physics. Since 2001, he has been a full Professor in Computational Sciences with the Informatics Institute of the Faculty of Science of the Universiteit van Amsterdam, Amsterdam, The Netherlands. His research focuses on the theory and application of complex systems through distributed mesoscopic computer simulation; trying to understand how information progresses through various spatial and temporal scales.