A Fuzzy Real Option Model for Pricing Grid Compute Resources

by

David Allenotor

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science
University of Manitoba
Winnipeg

Abstract

Many of the grid compute resources (CPU cycles, network bandwidths, computing power, processor times, and software) exist as non-storable commodities, which we call *grid compute commodities (gcc)* and are distributed geographically across organizations. These organizations have dissimilar resource compositions and usage policies, which makes pricing grid resources and guaranteeing their availability a challenge. Several initiatives (Globus, Legion, Nimrod/G) have developed various frameworks for grid resource management. However, there has been a very little effort in pricing the resources. In this thesis, we propose financial option based model for pricing grid resources by devising three research threads: pricing the gcc as a problem of real option, modeling gcc spot price using a discrete time approach, and addressing uncertainty constraints in the provision of Quality of Service (QoS) using fuzzy logic.

We used GridSim, a simulation tool for resource usage in a Grid to experiment and test our model. To further consolidate our model and validate our results, we analyzed usage traces from six real grids from across the world for which we priced a set of resources. We designed a Price Variant Function (PVF) in our model, which is a fuzzy value and its application attracts more patronage to a grid that has more resources to offer and also redirect patronage from a grid that is very busy to another grid. Our experimental results show that the application of the PVF has helped achieve equilibrium between users satisfaction measured as QoS and recovery of the infrastructure investment made by the providers. In the absence of pricing benchmarks, we setup Commodity Base Prices (CBP) and then integrated our PVF and CBP with GridSim to price grid compute resources.

In summary, this thesis provides the design of a model to price grid compute resources using financial options theory. The model achieves mutual benefit for users and providers in the grid environment. The mutual benefit is expressed in terms of QoS to the users and recovery of investments on the grid infrastructure for the providers. This thesis has opened up many different opportunities for further research especially in the era of enterprise computing with clouds.

Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgments

Forever I will sing the goodness of the Lord. Psalm 89:1

Rejoice in the Lord always. I shall say it again: rejoice! Philippians 4:4.

As I finish this thesis, I indeed cannot help rejoicing in the Lord - not only because I am finally done (which is a miracle in itself!), but even more because, as I look back on these past years, I recall all the ways God has blessed me, and it just fills me with gratitude and joy. In this short section, I would like to acknowledge the biggest of these many blessings - the people in my life who, in one way or another, whether they know it or not, have been instruments of God's grace and love for me throughout this long and lonely journey of PhD.

First, I would like to thank my advisor Dr. Ruppa K. Thulasiram for his kind support and help throughout the concluding part of my M.Sc. and for all the years of my PhD program at the University of Manitoba. I also want to express my profound thanks to my thesis committee members; Prof. Charles Mossman, Prof. Desmond Walton, Prof. Peter Graham, and Prof. Rajkumar Buyya. They made this thesis possible. Thank you for all your feedback, your ideas, and for your time. I would also like to give my very special thanks to Dr. Daniel O. Obasogie, my mentor and advisor (Diploma (E/Electronics)) at the Federal Polytechnic, Auchi. He was also my mentor and my advisor (B.Sc. and M.Sc. Computer Science) at the University of Benin, Benin City, Nigeria. Dr. Obasogie was first to teach me the fundamentals of Fuzzy logic design and digital control circuits. There is a little beginning to every great height of achievement in life. One of such humble beginnings is my journey to Canada for further studies which would not have been possible without the help of

Dr. S.A. Ehikioya. I appreciate the Ehikioya's family for their wonderful support and love towards my family and study in Canada.

Special thanks to my children: My daughter Ella (Kulukulu), and my two sons, David (Toopee) and Jesse (Anebhe-eEziza). I specifically thank my daughter, Ella who has not only been caring to me, but who has also unknowingly played a key part in this thesis – she always ask me "Daddy when will you be done schooling? Did your teacher give you too much home work? Can I help you?" Those words have put much challenge in me to get done quickly. Thanks also to my officemate at the University of Manitoba, Dr. Rodrigo Vivanco for his company and friendship. I also want to thank all the members of my labs; the Computational Financial Derivatives (CFD) Laboratory and the InterDisciplinary Evolving Algorithmic Science (IDEAS) for their ideas during the early stages of my PhD. research.

Finally, special thanks to the people back home without whom, I could not have gotten to the University of Manitoba. Thanks to the Head, Department of Computer Science, University of Benin, (Dr. S.C. Chiemeke) and the Vice Chancellor of University of Benin, for approving my study leave at the University of Manitoba. I thank all staff and colleagues at the department of Computer Science University of Benin. Also, I thank my very good friend Bamidele Ola. We came a long way and God came true for us in our M.Sc. days at the University of Manitoba. Others include the families of Sylvester Aghidi, Femi Olumofin, John Akinyemi, Christopher Iyogun, and the Nwani (Nnamaka, Kingsley, Nwanyor, and Ebele) family, and everyone in Canada (Winnipeg) who have positively influenced my study in Canada.

After all these years of my study (27 years in total from kindergarten), I have

come to realize that life is really not just all about research and work. In fact, if there is one lesson that I have learned in all the years of my Ph.D. education at the University of Manitoba, it is that life is a whole lot more than just research and work. Discovering new ideas, writing programs and papers, and going to conferences are all great, but in the end, these are all vanity compared to the more important things in life - how we love others, experience their love, and grow in God together. In this light, I would also like to thank all the people who have not necessarily contributed directly to this thesis, but who have, by their love and friendship, nevertheless made all this, and more, possible.

Most importantly, I would like to thank my childhood guardians/adopted parents, the families of Thuruthel Jacob (Ajith, Sarah, Sabeena, and John) and the Paul Jacob family (Mr. and Mrs. Paul Jacob and their Children, Ruth and Reuben). My tender age also was supported by the families of Chief Augustine A. Amune who was so proud of calling me a son. All these people filled my life with love, care, and support from the very beginning, and they have taught me virtues that keep me till date. They raised me up well, and taught me the important things in life. Without them, I would not be where I am, and who I am, right now. This thesis is every bit theirs as much as it is mine. I can only hope that as I now reach the summit of my academic education, I can continue in the rest of my life's education by learning to be as loving as they are.

I also thank my siblings: Mrs. Theresa Owobu, Mrs. Maggie Okonufua, Mr. Monday Allenotor, and Dr. Wilson Allenotor for your love, prayers, and support, and for making me enjoy and look forward to see you someday. I could not have done my

thesis without them. Thanks too to my uncle Chief Ehibor Oriarewo and family for prophesying good things to my life when he discovered my professional calling at a tender age. I thank also all other uncles, aunts, and cousins for their love, friendship, and support since I was little boy. Thanks to all my friends from back home, who have supported me with their prayers and friendship even from a distance. Special thanks to my best man Clement Okosun and Friday Ator (the rabbit-man).

Very special thanks to my brethren in the Redeemed Christian Church of God (RCCG) Winnipeg. They have been instruments of God's peace and love to me, and I have grown a lot because of them. Aside from my brethren in RCCG, God has also sent me good shepherds to strengthen, guide, and encourage me along my spiritual journey. First, I would like to thank my pastor, Pastor Joseph Okunnu and family for their love and encouragement during "my stormy sea" I also want to thank God for the life of our Pastor Akindele Odeshi for making me to stand in front of the congregation to speak to the people of God. Pastor Odeshi actualized God's desire for my life. He taught me the wonders and mysteries of God's love, kindness, and endurance through the grace of God. Pastor Odeshi has done so, not only through his kind and wise words in teaching the gospel, but even more through the example of his life.

Finally, I would like to thank my dearest friend, and God's most wonderful gift to me, my wife, Anna. She has been with me through the hardest times of my stay at the University of Manitoba and in Canada, and has made these times also the happiest. My life has truly been so much more joyful since I have met her, and I feel so blessed and thankful to God for blessing us with three children while I was doing

my PhD research.

As I end here, I would like to thank God again, who has given all these blessings, and without whose grace none of this would have been possible. It always amazes me how God's plans for us are so much bigger and better than we can ever imagine ourselves. So now, I look forward with much joy and anticipation to what He has planned next for my life *Ephesians 3:20 "Now to him who is able to do immeasurably more than all we ask or imagine, according to his power that is at work within us"*

Thank you so much, Lord!

I dedicate this thesis to the loving memory of my late parents "Abba ogie" Dr. Ateṣi Allenotor (who left me to be with The Lord three months after I was born) and my mother Mrs. Ibhanabhor Allenotor (ododo nee rẹ guẹgbẹ), and to the loving memory of Mr. and Mrs. Thuruthel Varkey Jacob. ad majorem Dei gloriam. Peaceful sleep until we meet and stay together forever.

Chapter 1

Introduction and Motivation

One of the key trends in today's computing and Information Technology (IT) is service-oriented computing. Service-oriented computing is entirely driven by distributed computing technologies such as Service-Oriented Architecture (SOA) which is similar to the grid computing idea. Grid computing refers to the sharing of computing resources such as CPU cycles among a set of distributed computers. A computational grid is analogous to an electrical power grid [2] with a common vision to offer dependable, consistent, pervasive, and inexpensive access to high-end resources irrespective of their location to users. The origin of the grid concept may be traced back to the late 1990s [4] with the need to deploy massive gigabit testbeds such as Collaborative Adaptive Sensing of the Atmosphere (CASA) [5] to link super-computing sites across the United States. There are various definitions for the grid, however, Foster et al. [2] and Buyya and Venugopal [6] describe a computational grid as a parallel and distributed system (software and hardware infrastructure) that enables the sharing, selection, and aggregation of geographically distributed autonomous resources

dynamically at runtime depending on their *availability*, capability, performance, *cost* (resource pricing), and user *preferences* (e.g. users' satisfaction guarantee). The major objective that a grid computing paradigm strives to achieve is the provision of shared compute services or resources [2] to users with a high Quality of Service [7] (QoS).

A well known example of a grid computing project is the Intel's NetBatch Grid project uses spare computing power available in Intel's various offices to process engineering simulation jobs. Intel estimates that NetBatch has saved the firm $50M over 10 years and increased computer utilization by 45% over the same period [8]. Grid computing offers great benefits in solving computationally complex problems such as those related to financial modeling, weather prediction and molecular modeling for Biotech applications. As a result, there has been a considerable increase in the use of grid resources. Several firms including Sun, IBM, and Amazon have expended considerable research effort towards offering grid resources as services. For example, Sun Microsystems recently launched the Sun Grid where users can submit large jobs, and charged they are $1.00 per CPU hour. IBM runs three grid centers in the US and one in France; they currently charge $0.47 per hour for CPU usage. The resource usage charging scheme used by these initiatives is static, and in several instances they are profit-driven. Similarly, Amazon's Simple Storage Service (S3) and Elastic Cloud Compute (EC2) now charge for use of grid resources. Section 1.1 decribes the charging scheme used by Amazon and other commercial grid resource providers.

The term "service" as used in a grid computing context, in general, and in this thesis in particular, describes the provision and exchange of information/data made

available to a subscribed user by a Grid Service Provider (GSP). In the context of this dissertation, quality is used to describe the degree of satisfaction of service to a subscribed user's expectations in a computational grid. Therefore, a user's perception of a service to a set of predefined service conditions (service agreements) necessary to achieve the specified service quality is described as quality of service and measured by QoS[1] and these service agreements are contained in the Service Level Agreement (SLA) [7]. A grid QoS-SLA document describes the contract relationship between the grid resource provider and the resource user. Often times, the user specifies the expected level of service during the terms of the contract (see Appendix C for a typical SLA template). The SLA document protects the grid resource provider and user by ensuring that they adhere to the terms of the contract. However, several factors hinder grids in providing effective service.

First, a grid is a capital-intensive infrastructure to own as an individual. As a result, ownership of the grid resources (CPU cycles, memory, network link bandwidths, disks, various visualization tools, software, and specialized instruments – which I call grid compute commodities gcc-s) come from different domains which makes access policies as varied as the owner organizations.

Second, the grid has a resource availability issue. In a grid, the gccs are available in a non-storable fashion i.e. as compute cycles. This means that a resource that is available at one time may become unavailable at other times. This problem of grid resource availability is one of the major issues since the gccs are federated from geographically distributed sources with different operational and access polices. Non-storable characteristics of grid resources also affect the overall Quality of Service

[1]QoS is a fuzzy expression of the level of user satisfaction.

(QoS) for the reason that the grid QoS is a global function of the individual local Service Level Agreements (SLAs) and it is hard to create a globally consistent SLA.

Third, Information Technology (IT) is continuously advancing. For example, changes such as newer technologies, faster algorithms or improved devices may result in a huge change (a drop in the projected resource utilization or a cancelation of initially booked service) in grid user's behavior. This behavior could be attributed to the fact that most of the applications that run on the grid are pre-budgeted by way of an indication to use the resources in the future. Therefore, the design of underlying grid middleware architecture should incorporate the dynamics that can handle such changes. Similarly, most businesses find it difficult to adjust to rapid changes (since adjusting to changes most often result in extra operational costs) in IT such as regular upgrades in applications, hardware, and other specialized tools that additional investment in IT infrastructure. The objective of my dissertation is to price grid resources under these complex situations. To achieve this objective, I treat *gcc* as real assets and employ the following techniques:

1. Financial Options – since grid resource availability is transient and usage is based on availability, I treat the grid resources as compute commodities and apply the theory of financial option pricing to price them. The major objective that a grid computing paradigm strives to achieve is the provision of shared compute resources [2] to users at a high QoS. Other sub-objectives include the provision of grid resources to users at prices significantly lower than the separate market cost and at the same time ensure that the provider recovers the investment on the grid infrastructure. One of the ways to achieve these objec-

tives is to increase grid patronage. This will keep the grid busy and guarantee a faster rate of cost recovery on infrastructure expenses incurred by the providers. However, it is hard to achieve these objectives in the presence of flexibility for the use of grid resources, which is characterized by user opportunities from the decisions to utilize the resources. Such flexibilities can be accurately captured using real options so that asset prices could be computed using financial options pricing techniques.

2. A discrete time trinomial lattice technique is employed to compute price (option value) for the gccs. This step in pricing is done by formulating the grid compute commodities pricing problem as a real option pricing problem. I apply a "real option" approach to capture uncertainties that abound in making the decision to exercise the option for using grid resources.

3. Fuzzy Logic is used to capture the uncertainties in the provision of user QoS.

Since currently there are no benchmarks for pricing grid resources using financial options, I apply the principles of Activity Based Costing (ABC) [9] to compute the resources' base prices by taking into consideration infrastructure cost only. Other overhead costs (for example building, mortgage, equipment insurance, salaries, equipment upgrades, heating, cooling etc.) are not considered in this thesis.

1.1 Commercial Grid Resource Providers

Due to the cost of system upgrades, maintenance, and a rapid depreciation of computing infrastructures, various IT businesses (e-business and e-commerce activities)

prefer to rent resources (services) instead of buying them for computation purposes. Several companies are now offering access to resources as a service. Some of these companies include: AppNexus [10], GoGrid [11], Joyent [12], RackSPace [13], Google Application Engine [14], Amazon Web Services (AWS) [15] and Microsoft Grid [16]. Amazon Web Services (AWS) include the Simple Service Storage [17] (S3), the Elastic Compute Cloud (EC2), and the Simple Queue Service [18] (SQS). EC2 is a Web service that provides access to resizable compute capacity or Web-scale computing. The related Amazon Simple Storage Service (Amazon S3) is a cloud storage service that provides storage on demand [18]. Cloud computing is a system that involves dynamic scaling of virtualized resources which are provided as services (storage, platform, computing power, application, software, and hardware) – resource offered as a service (Resource as a Service (RaaS)) over the Internet.

Amazon EC2 is a Web-based computing service that provides resizable compute capacity [19]. It offers on-demand computing resources in the form of a virtual machine that is accessible via the Internet. Using Amazon EC2, a user has full control of virtual machines equivalent to a 1.7GHz Xeon CPU, 1.75GB RAM, 160GB HDD, 250Mb/s network at a price of $0.10 per instance-hour (or part hour) [18]. Amazon's EC2 uses the XEN [20] virtual image platform to offer an on-demand operating systems of choice that provides a complete virtual computer with a CPU, memory, and disk space. The pricing scheme offered by Amazon is simple. Charging is done per instance hour used. It is not possible to apply dynamic prices or reserve computational power [19]. Tables[2] 1.1 and 1.2 show the per hour static prices (as of June 2010) of an Amazon EC2 instance machine in the United States and Europe respectively. An

[2] 1 Month = 4 weeks ($24 \times 7 \times 4 = 672$ hours).

Amazon instance machine could either be a 32bit or a 64bit machine. For instance, the first entry in Table 1.1 corresponds to a small 32 bit system with 1.7GB of RAM and 160GB Hard Disk space. This machine could run a Linux Operating System (OS) for $0.10 per hour (ph) or a Windows OS for $0.125 ph. Similarly, Table 1.2 shows a small 32 bit instance machine of the same configuration but for the Linux OS the cost of use is $0.11 per hour (ph) or for a Windows OS, $0.135 ph.

Table 1.1: Instances of EC2 in the United States

	RAM	**DISK**	**LINUX**	**Windows**	**SQL Server**
Small$^{(32bit)}$	1.7GB	160GB	$0.10 ph $72 monthly	$0.125 ph $90 monthly	
Large$^{(64bit)}$	7.5GB	850GB	$0.40 ph $284 monthly	$0.50 ph $360 monthly	$1.10 ph $792 monthly
Extra Large$^{(64bit)}$	15GB	1690GB	$0.80 ph $568 monthly	$1.00 ph $720 monthly	$2.20 ph $1584 monthly
High CPU$^{(32bit)}$	1.7GB	350GB	$0.20 ph $142 monthly	$0.30 ph $214 monthly	
High Extra CPU$^{(64bit)}$	7GB	1690GB	$0.80 ph $568 monthly	$1.20 ph $864 monthly	$2.40 ph $1728 monthly

Table 1.3 shows a comparative summary of monthly resource costs for six companies that provides resources as services.

1.2 Definition of Terms

Financial options, real options, fuzzy logic, and the basics of cost accounting principles are the four research threads that form the background of this thesis. This section presents definition of terms in the context of their meaning for this thesis.

Table 1.2: Instances of EC2 in Europe

	RAM	**DISK**	**LINUX**	**Windows**	**SQL Server**
Small$^{(32bit)}$	1.7GB	160GB	\$0.11 ph \$79 monthly	\$0.135 ph \$97 monthly	
Large$^{(64bit)}$	7.5GB	850GB	\$0.44 ph \$316 monthly	\$0.54 ph \$388 monthly	\$1.14 ph \$792 monthly
Extra Large$^{(64bit)}$	15GB	1690GB	\$0.88 ph \$633 monthly	\$1.08 ph \$777 monthly	\$2.28 ph \$1584 monthly
High CPU$^{(32bit)}$	1.7GB	350GB	\$0.22 ph \$158 monthly	\$0.32 ph \$230 monthly	
High Extra CPU$^{(64bit)}$	7GB	1690GB	\$0.88 ph \$633 monthly	\$1.28 ph \$921 monthly	\$2.48 ph \$1785 monthly

Table 1.3: Cost Comparison

	Amazon	**GoGrid**	**FlexiScale**	**Mosso**	**ElasticHosts**	**Joyent**
Memory	7GB	8GB	8GB	15GB	8GB	32GB
Hard Disk	1690GB	480GB	100	620	1862GB	100GB
Cost/Hour	\$0.08	\$.037	\$0.53	\$0.096	\$0.76	\$5.95
Cost/Month	\$537.60	\$255.35	\$358.50	\$645.12	\$510.72	\$4,000

Each of the terms is defined in relation to resource pricing.

1.2.1 Financial Option

A financial option is a contract that gives the right to its holder to exercise the option. That is, a financial option (see, for example [21]) gives the right to buy or sell an asset (for example, a stock) under certain future terms for a given period. Since the holder has rights but without obligations, the option has value. An option

is a derivative because its value depends on the value of another asset which is called the underlying asset. Examples of underlying assets are stocks, foreign currencies, stock indices, debt instruments, futures contracts, and commodities. In the case of real options (Chapter 5), the underlying assets include expected results of cash flow from projects (for example, the expected preset cash flows from an investment in grid resources).

Two types of options are call options (calls) and put options (puts). A call option gives the option holder the right to buy an underlying asset by a certain date for a certain price. A put option gives the option holder the right to sell an underlying asset by certain date for a certain price. The option holder may decide to use or not to use the option. If the holder decides to buy/sell the underlying asset using the option, it is said that the option is exercised. The price at which the asset can be bought or sold is called the strike price or exercise price of the option. The date when the contract expires is known as the maturity or expiration date. These two types of options are available in many different styles. A European option can be exercised only on the maturity date, while an American option can be exercised any time up to the maturity date. The writer of the option gets the price of the option or the option premium, when the contract is agreed upon and the writer accepts potential liabilities in the future.

If the asset price S is less than the strike price K of the option, the holder of a call option may not want to exercise the option because the same asset can be bought from the market at a lower price. However, whenever S is greater than K, the call option can be exercised and the holder of the option can make some profit

equivalent to $(S - K)$ since the holder can buy the underlying asset for a price at K and sell immediately at price S. The difference between the asset price and strike price at maturity is generally referred to as the pay-off. At any time until maturity, the difference between asset and strike price is called the intrinsic value of a call option. This would be the pay-off if the option was exercised instantly. If C denotes call option and P denotes a put option, then the general formula for the value of a call option C and a put option P at maturity [21] is given as:

$$\begin{aligned} C &= max(0, S - K) \quad \text{or} \\ P &= max(0, K - S) \end{aligned} \tag{1.1}$$

Equation (1.1) implies that higher asset prices give higher values for a call options. For put options, it is just the opposite. The length of time to maturity affects the option value. For example, for an American option, a longer time to maturity increases the option value because the holder of the option has all the exercise opportunities open through the life of the option. For a European option, the effect of time to maturity is ambiguous. However, a longer time to maturity increases the value of a European call option, where there are no dividends during the life of the option.

1.2.2 Real Option Theory

Real option analysis provides evaluation schemes for major large projects including assessments, Information Technology projects. Real option is advantage over the traditional quantitative capital budgeting techniques such as various Discounted Cash Flow (DCF) methods including Net Present Value (NPV), and Internal Rate of Re-

turn (IRR) is seen in its ability to recognize management flexibility ([22] and [23]. These managerial flexibility include the choice to execute a project, abandon a project or wait for a better time. To evaluate world wide Information Technology spending, estimated to be around $3.00 trillion [24] in 2008, NPV, DCF, and IRR will fail to capture the estimated spending because of the inherent uncertainties involved. Real option theory provides a mechanism for valuating investment opportunities in situations where the cash flows are not deterministic. Real option application can also be useful when quantifying the accrued gains from prototyping a product as well as in managerial flexibility. Several of these gains can ordinarily be hard to quantify [25]. As a result, a comparison of alternative opportunities becomes harder to evaluate whenever managerial flexibility varies across the opportunities. This is the case with a computational grid system. A grid's ownership, policies (managerial/addministrative), and operating functions vary across wide geographical zones. Given such variabilities in the grid, traditional valuation schemes such as DCF and NPV are unsuitable for use because the NPV and DCF analysis ignore the effects of uncertainty. For example, grid resource use requested at a time t, may be for immediate use when $t = 1$, or in the future (if t is large positive). So, I say flexibility abounds in the decision to use the resource. The presence of these flexibilities grants the user an obligation-free grid resource usage. If the users' computing needs change in the future, the user may modify the requests to reflect the anticipated usage. Therefore, to price grid resources in the presence of these flexibilities, I treat the Grid Compute Commodities (gcc) as real assets and I employ three steps: (a) I model the pricing function as a real option problem by formulating the grid compute commodities pric-

ing problem as a real option pricing problem, (b) I model the grid resources spot prices using a discrete time approach, and I apply a trinomial lattice with multi-asset gccs in the option and (c) I address uncertainty constraints inherent in achieving required quality of service (QoS) and grid resources availability using fuzzy logic. I have also introduced a PVF (Section 6.2) to achieve the objective of pricing the grid resources.

1.2.3 Fuzzy Set Theory

Zadeh [26] introduced fuzzy set theory as a technique that models uncertainty (impreciseness) in natural language. In the traditional (crisp) set theory, an object has two distinct possible membership values: Either the object is a member of the set or it does not belong to the set. Therefore, a logical proposition either holds or does not hold for a traditional set. In fuzzy set theory, a membership function describes the degree to which an object belongs to a fuzzy set. Let X be a non-empty set; then a fuzzy set A in X is characterized by its membership function given as:

$$\mu_A(x) : X \rightarrow [0, 1] \tag{1.2}$$

where $\mu_A(x)$ is called the degree of membership of the element x in fuzzy set A for each $x \in X$, that is,

$$A \ = \ (x, \mu(x))|x \in A, \mu_A(x) \in [0, 1] \tag{1.3}$$

A membership function $\mu_A(x)$ specifies the grade or degree of membership to

which x belongs to the fuzzy set A. The mapping in Equation (1.2) associates each element in the fuzzy set A to a real number. Thus, as $\mu_A(x)$ tends to 1, the degree of membership function tends to 100%. The choice of membership function is application dependent [27]. Membership functions have many shapes that depend on the fuzzy data set. The most common shapes of functions are triangular, trapezoidal, bell shaped, and parabolic. In this thesis, the trapezoidal membership function is used for its simplicity [28] in the analysis of fuzzy logic systems. It can be considered an extension of the triangular membership function. The general form for the triangular membership function is given in Equation (1.4) as:

$$\mu_{\Delta}(x) = \begin{cases} 1 & \text{for } x = b \\ \frac{x-a}{b-a} & \text{for } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{for } b \leq x \leq c \\ 0 & \text{otherwise i.e., for } x \notin [a, c] \end{cases} \tag{1.4}$$

where $[a, c]$ denotes the universe of discourse and $b = (a + c)/2$ is the value for which $\mu_{\Delta}(x) = 1$. Figure 1.1 shows a triangular membership function. The general form of

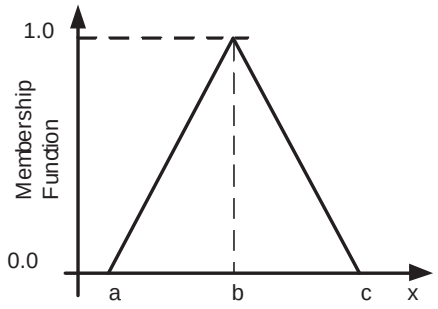

Figure 1.1: Triangular Membership Function.

the trapezoidal membership function is given in Equation (1.5) as:

$$\mu_{Trap}(x) = \begin{cases} 1 & \text{for } x \in [b, c] \\ \frac{x-a}{b-a} & \text{for } a \leq x < b \\ \frac{d-x}{d-c} & \text{for } c < x \leq d \\ 0 & \text{otherwise i.e., for } x \notin [a, d] \end{cases} \tag{1.5}$$

Figure 1.2 shows a trapezoidal membership function. In Figure 1.2, $[a, d]$ marks

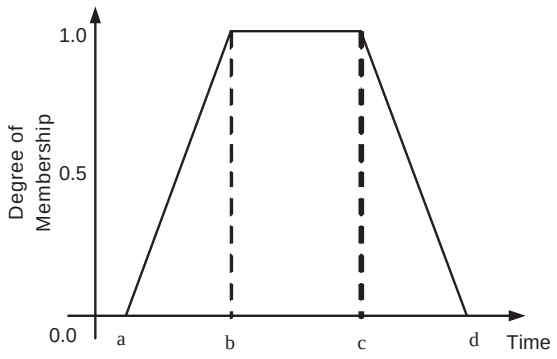

Figure 1.2: Trapezoidal Membership Function.

the universe of discourse, b and c are the values of X between which $\mu_{Trap}(x) = 1$. Readers are referred to [28], [29], and [30] for discussions on other forms of membership functions.

1.3 Problem Statement

Grid computing became popular in the late 1990s. One of its major advantages is in creating the ability for individual users to access personal grid computing infrastructure. Existing grid infrastructures are owned and funded by corporate bodies

or government organizations that make the resources available to the users (for research and academic purposes only) free of charge. Consequently, several application areas such as Science, Engineering, and Business, (where computational needs exceed their locally available capacity) run their applications on the grid. Currently there is over 80% grid resource utilization in production environments [31]. As a result of no cost to users due to government funding there are no efforts towards pricing the grid resources. Research efforts have focused mostly on issues such as security [32], middleware and grid infrastructure [2], advanced reservations ([33], [34], and [35]), and Grid resource management and scheduling [36].

In recent years, grid computing has become a popular means for executing computationally intensive and resource-intensive applications. The characteristic nature of the computational grid makes it difficult to price the resources using a few available economic models such as Discounted Cash Flow (DCF) and Net Preset Value (NPV) [25]. The DCF analysis represents the NPV of the projected cash flows available to all providers of capital (investors), net of the cash needed to be invested for generating a projected growth. The concept of DCF valuation is based on the principle that the value of a business or asset is inherently based on its ability to generate cash flows for the investors. As a result, the DCF relies on the expectations of the business i.e. on a set of assumptions. The accuracy of the valuation determined using the DCF method is highly dependent on the quality of the assumptions. In real projects such as the grid investments, uncertainties which were unplanned for may set into the project development. In DCF a business owner (such as a grid resource provider) who wants to invest in grid infrastructure must need to first determine the

variability of the grid investment (i.e. determine how many users can be attracted to the grid). Based on this information, the provider comes up with a static price. This type of pricing has only one opportunity opened to the provider – to make profit without a corresponding choice for the user.

In the context of grid resources utilization, pricing the grid resources is a challenge because any model that prices grid resources must also resolve uncertainties such as resource availability, guarantee QoS while considering grid resources that are owned and operated by geographically dispersed owners. Such uncertainties may not be captured accurately using traditional DCF or NPV models which do not account for the real value of the grid compute commodities. I have proposed in this thesis a finance option model to price grid resources which will achieve the following benefits. To the provider, it will help to recover the investment of the grid infrastructure and on the user side, it will provide the resources at a guaranteed QoS.

To price grid resources, I need a pricing scheme that guarantees resources availability and QoS (satisfy a user's computing needs). I propose a real option valuation pricing architecture to price grid resources. I develop my model using financial option theory from a real option perspective and value the grid resources by treating them as real assets. This approach is novel and distinct from existing economic models that apply a generic and traditional valuing methods such as DCF and NPV which does not account for the real value of the grid compute commodities. I apply a fuzzy real options valuation method to address QoS, a hybridized solution scheme that combines the advantages of both fuzzy logic reasoning and real options of a decision-based system to develop the pricing architecture.

1.4 Solution Strategy

The basic idea of this thesis is to price resources of a grid computing system. In an initial naïve approach, I collected compute resource utilization data for an Education grid (WestGrid), observed the usage patterns and analyzed the collected data. Figure 1.3 shows the Canada grid – Canarie and WestGrid. WestGrid is one of Canada's largest High Performance Computing (HPC) consortium. The Canarie project uses CAnet 4 and the regional networks in British Columbia (BCNet) and Alberta (Netera Alliance) to connect more than 1,000 researchers across Canada to grid-enabled HPC and collaboration infrastructure located at institutions in Western Canada. My results in a preliminary study [37] concluded that grid resource availability and guaranteeing QoS are two important factors that challenge the development of pricing models. Therefore, I suggested a mix of the application of real option and financial option theory to overcome the challenges I observed. Basically, I applied economic principles that are based on grid activities associated with resource usage and assigned cost as a result of the entire grid overhead; I apply fuzzy logic to capture uncertainties in the availability of grid resources; real options to capture the realistic value of the priced resource; and financial option pricing to compute the option value of the resources. My approach has the following general expectations: (i) my intended pricing model should be generic (independent of the environment of the grid – does not depend on whether the grid's environment is educational, commercial or experimental), (ii) the data collected should not bear any relationship to pricing using financial options.

The core idea of this thesis is to price grid resources and to strike and maintain

Figure 1.3: Canarie Network [1].

equilibrium between service satisfaction of grid users and cost recovery for service providers. As a user identifies the resources of a grid that are required for a computation, the user places a request based on the budget and time of usage constraints. The compute resources assigned for computations are selected based on a scheme similar to a reverse Dutch auction to price the grid resources in my simulation as described in Section 5.4.

Extensive simulations were carried out using a grid simulation tool called GridSim. I collected data in the form of actual traces from seven real grids: one commercial grid (Grid3 [38]), two experimental grid platform (Grid5000 [39], and LCG [40]), two research grids (Shared Hierarchical Academic Research Computing Network (SHARCNet) [41] and Western Canada Research Grid (WestGrid) [42]), Auvergrid, NorduGrid [43]; Then I applied my pricing model to these grids. I compared the computed prices of utilized grid compute resources across various grids in simulation and in experiment with real grid data. The results of my simulations indicate that the user and the provider of the grid resources benefit (in terms of early recovery on investments of infrastructures) from using my grid resources pricing model, with the option to fine tune the model towards profitability of providers.

1.5 Contributions

The contributions of the research presented in this thesis are both specific and general. The general impacts of the contributions are as follows:

1. The thesis proposes a resource pricing model that improves resource utilization and justifies information technology investments in a grid.

2. The thesis proposes a generic plan for managing the resources/infrastructure of a grid that is essential for meeting the peak demands of grid resources utilization.

3. To the government and funding agency that funds the grid, this thesis contributes in the following ways: (i) It will justify government spending and quantify the services provided in terms of actual value and (ii) It will provide

effective management of the distributed resources.

The specific contributions of this dissertation are in the area of grid resource pricing using finance option theory and are:

1. As a resource pricing architecture, this thesis uses a modified reverse Dutch auction to price the grid resources. Using this procedure, the user is able to set usage preferences according to available budget and time deadlines. Two advantages of this are (i) a system that guarantees user's QoS for resource utilization and eliminates the possibility of increasing the cost of computation for jobs that have lower waiting times[3] and (ii) a system that keeps the grid busy for optimal gain (user and provider "profitability"); this will avoid idle states in the grid.

2. The model will also guarantees service and satisfaction in terms of the QoS requirements for grid resource users and resource owners through a regulated Service Level Agreements (SLAs)-based resource pricing infrastructure.

3. Design of pricing architecture.

4. My model uses the in-built flexibility available in option theory (real option based) for pricing the grid resources by providing a handle of control for both users and the provider benefits; the users's QoS is not compromised while the provider achieves the benefits of early infrastructure investment recovery.

5. My incorporated trinomial lattice with the pricing model shows a bridge between the pricing grid resources and gap

[3]One of the shortcomings of EGEE (see Section 2.2.5)

6. Adapting and implementing of my pricing model with GridSim simulator demonstrates the possible adaptation of my model in a real grid middleware which will open up more research in the area of grid resources pricing.

7. The design of commodity base prices in the absence of standard pricing bench marks and the analysis I made using real grid trace data shows that my model can be used to achieve the desired purpose of pricing grid resources in grids.

1.6 Summary of Knowledge Advanced

Following a review of related work in financial options and grid middleware frameworks, I collected and analyzed resource usage and users' behaviors in a computing environment (Polaris parallel machine the University of Manitoba) in a one year period. The observations from the utilization pattern indicates that grid resources pricing can be modeled using financial options. Data collection was extended to WestGrid High Performance Computing (HPC) consortium (as a case study). Following the usage analysis, as a proof of concept, I developed the idea of an initial model using hypothetical data on a trinomial lattice (see Section 3.1.2 for model assumptions). Important results at this stage of research showed that the price of a grid resource is determined by the effects of the exercise time of the option to use the grid resource even in the presence of uncertainties that are hard to determine using conventional techniques such as Net Preset Value (NPV) or Discounted Cash Flow (DCF). This finding lead to the deduction that grid resources may be priced using real options as a tool. The initial findings are published in [37].

The extension that involves more than one grid type was the next stage in the thesis workflow. This stage used data from a grid other than the WestGrid – the Shared Hierarchical Academic Research Computing Network (SHARCNet) [41]. This led to the development of a conceptual and formal middleware architecture. This stage emphasized increased profitability and imposes stringent conditions for the users QoS and I applied fuzzy logic to model the users' expected QoS. However, I observed that in both SHARCNet and WestGrid, sometimes the number of jobs supported (the grid patronage) significantly decreases and makes the grid idle (some periods in a given year). A low patronage makes it hard to guarantee the providers profitability. Therefore, I need to increase patronage by applying a price controller into my model. Following this observation, I proposed an initial idea of a price adjustment system. Results at this stage led to the development of the Price Variant Function (PVF or p_f) [44] to control the large fluctuations in grid resource utilization. The PVF is an important function of my model and its based on advances in technology. The design of PVF in a multi-grid multi-resource system with a very large numbers of users could be a hard problem. However (in my future work), the application of input-output Economics [45] when applied will reduce the complexity involved in a large system into an inputs and outputs interaction.

In what followed, grid resources usage trace data were collected from five other real grids to test feasibility of PVF in my model (see Chapter 1 for a detailed description). Since there are no benchmarks for resource base prices, I choose some reasonable and justifiable base values for an initial test in [46]. Several experiments have been carried out to show the application (control of fluctuations in the resource prices) of the PVF

in [47].

1.7 Thesis Organization

The rest of this thesis is organized as shown in Figure 1.4 where each shaded block represents a "module". The introduction module consists of two chapters. Chapter 1. The thesis introduction and general overview are motivated in Chapter 1 while the terminology, definitions, and a description of the problem to be solved are explained in Chapter 1. The literature review module presents work related to the current research. Related areas to the current research which I reviewed in Chapter 2 include financial options, real options, fuzzy logic, grid computing, grid services, economics of grid marketplaces, and cost accounting. Chapter 3 is contained in a separate design module called the basic design (methodology) module. In Chapter 3, the basic model design is presented using trace analysis (phase one and phase two) obtained from two Canadian academic and research grids (SHARCNet [41] and WestGrid [42]). This stage of my design is the initial and basic model which was refined in the integration and simulation module discussed in Chapter 4. In the simulation and integration module, I review how I integrated the basic design into GridSim and applied financial option pricing theory. A refinement of the results after simulation was also carried out in the results and discussion module. The results and discussion module consists of two chapters: Chapter 5 and Chapter 6. Chapter 5 includes trace data from five another grids (Grid5000 [48], AuverGrid [49], DAS-2 [50], LCG [40], NorduGrid [43]) were analyzed. The results of the trace analysis and the resulting refinement are discussed in Chapter 6. The conclusion module consists of Chapter 7. In this chapter,

I provide the research conclusion and future research directions.

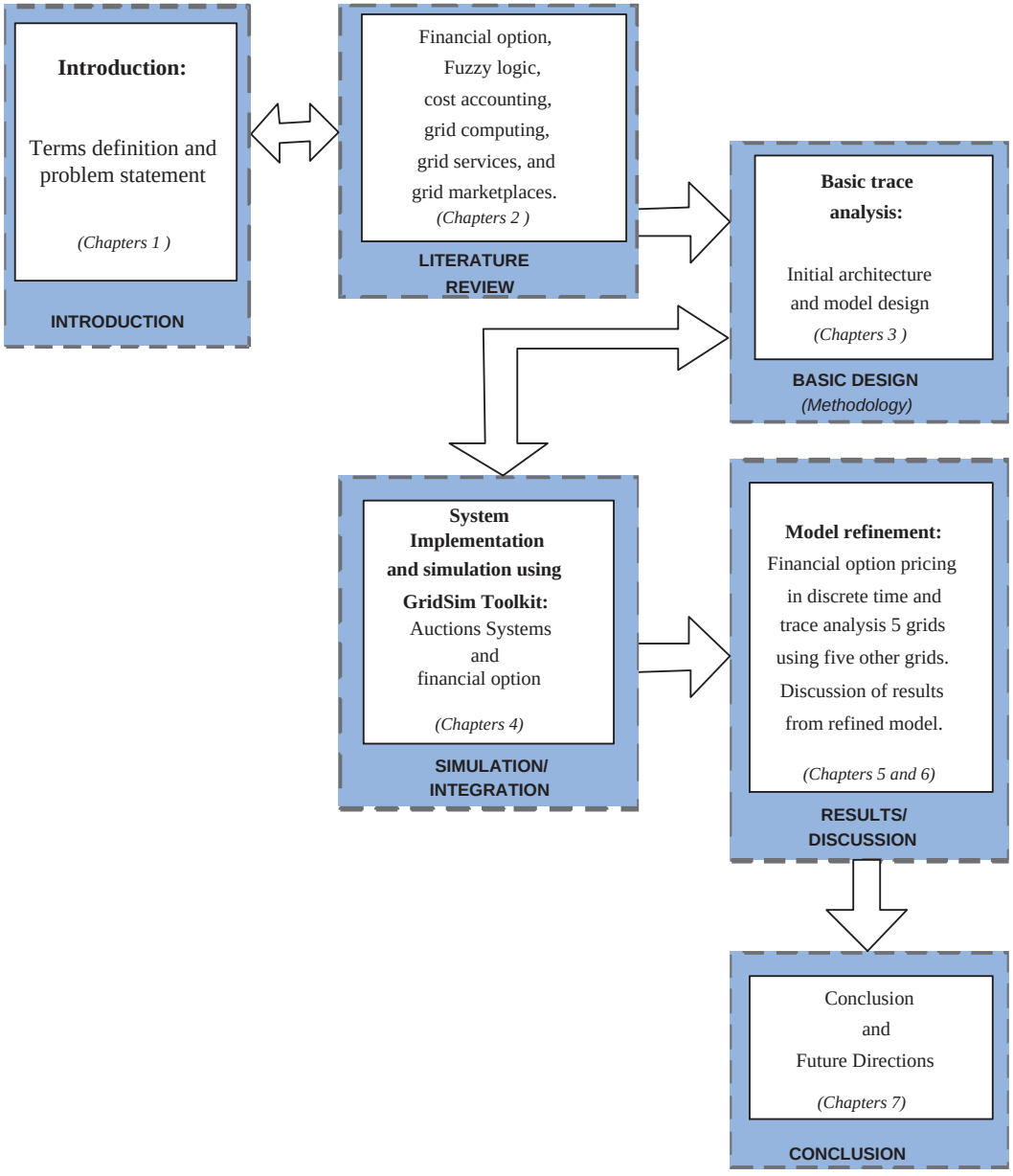

Figure 1.4: Thesis Organization.

1.8 Chapter Summary

In this chapter, I provided the essential motivation for grid computing to the general reader while exploring the architecture of grid middleware. I also examined existing commercial grid resource providers and their charging schemes. The non-storable characteristics of grid resources make it difficult to price them. I then conclude that grid resource pricing should involve provisions than assigning cost of usage. I identified other parameters to associate with simple charging, such as the ability to guarantee resource availability measured as QoS. In this chapter I identified that the use of conventional methods such as the DCF or the NPV cannot guarantee effective measurement of the characteristics value of the priced grid resource, and hence motivated the objective of this thesis. I defined some of the terminology that I will use throughout the thesis. I provided some common definitions associated with financial options. I also described the essential principles of fuzzy set theory and the mathematics of fuzzy modeling.

Chapter 2

Literature Survey

This chapter provides a review of related research in grid computing, resource management, financial options, real options, and resource pricing. The chapter also examines the relationship between the related work understudy and resource pricing in a grid market.

2.1 Grid Computing Related to Pricing

Besides the focus on security related issues [32], middleware and grid infrastructure [2], most of the current research related to grid resource pricing focuses on market based economy approaches (for example [51], [52], [53], [54]) and contingent bids in auctions [55]. These research efforts use economic principles to decide a fair share of grid resources that involves resource redistribution and scheduling. Currently, there is no charge for using grid resources. However, a trend is developing for public computing using a grids. Therefore, a sudden explosion of grid use is expected in near

future. Iosup et al. [31] analyzed and compared the resource utilizations of four Grids, one academic and three production. Their analysis was seen from both a user and a system perspective. They observed that despite a high percent of grid applications employ a parallel model, grids were not yet utilized at their full capacity. They suggested that grid resources use will reach a peak value soon and this could lead to grid capacity problem [31] whereby the grid will have more jobs than it can handle or will grant resources to users while compromising the users' QoS. To address the problem of sudden explosion in the use of computing resource, Amazon's Simple Storage Service (S3) [17] now offers a pay-as-you-go online storage. The S3 has gradually developed into EC2 and provides an alternative to in-house mass computing.

Several other research efforts explore the possibilities of incorporating resource pricing into the grid infrastructure. These include Kang et al. [56] and Tan and Gurd [57]. Researches in grid followed two distinct approaches; they either extend the existing standardized grid middleware or to see pricing problem as a resource allocation and resource management issue. However, resource management is not only about scheduling large and compute-intensive applications (or resources), or some form of advanced reservations [58], [34], [35]. Resource management also involves the manner of putting compute resources to work for the benefit of the user and owner [44]; that is, service for the users and profitability for the provider. Similarly, Chunlin and Layuan [59] presented an optimization-based resources pricing algorithm. They focus on resource sharing in a computational grid while they keep increasing the grid providers' effective gain. In another related study on grid resource valuation, Sulistio et al. [60] evaluated the effectiveness of grid revenue management using resource

reservations as a focus. They show that charging customers with differentiated prices will increase the effective total revenue for the resource in question. They also showed that their scheme guarantees a fair share of the resources applications with highest computing priorities.

Kang et al. [56], Tan et al. [57], and Sulistio et al. [60] focus on resource sharing and resource scheduling with specific reference to a market economy. Mutz et al. modeled job priority in a batched queue system using the efficient design mechanism previously by Krishna and Perry in [61] based their proposal on a compensation function that schedules jobs for a time t_i. Mutz et al. concluded that a compensation function d should be paid by n given job at t_{n-1} that wishes to access computation at t_n. The compensation may be disbursed as incentives (say more gccs) to jobs willing to wait.

Bhargava and Sundaresan [55] model a computing utility and examine the possibility to extend the pay-as-you-go pricing using an auction system. Their proposal is limited by some assumptions. They assume that the participants operate under risk neutral conditions; i.e., a bidder that fails to participate in a bidding process gets a refund. In a recent study [47], I focused on balancing the grid profits as seen from the perspectives of the grid resources provider. I use (i) an option with dividend (dividend is treated as incentive) paying the underlying asset and (ii) a penalty function – the PVF. (Section 6.2 provides details).

2.2 Grid Middleware

Foster and Kesselman [62] describe a grid middleware as an architecture that consists of fundamental system components such as the application layer, the collective layer, resource layer, connectivity layer, and the fabric layer. A grid architecture consists of a set of protocols that defines the basic mechanisms by which virtual organizations, users, and grid resources negotiate to establish, and manage sharing relationships in a grid. The middleware specifies the purpose and function of the components and indicates how the various components interact with each other. Figure 2.1 shows the layers required to form a grid as described Foster et al. in [2].

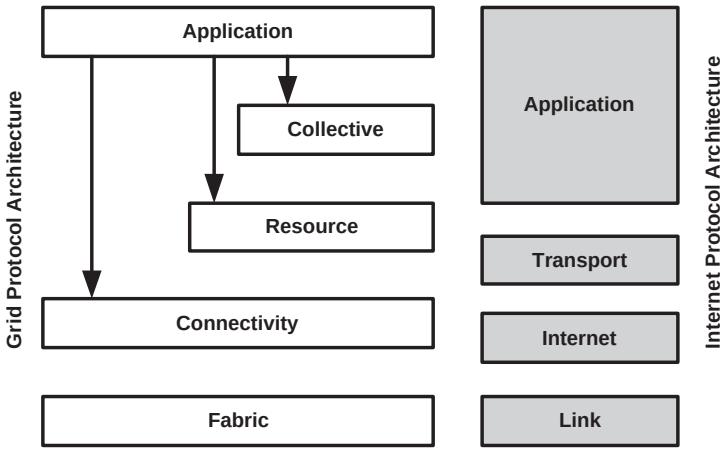

Figure 2.1: Grid Layered Architecture [2].

According to Foster [2], the grid is organized into five layers which include a fabric layer, the connectivity layer, resource layer, collective layer, and the application layer. The fabric layer is located at the lowest level of the grid architecture. It consists of the shared resources such as computational resources, storage systems, catalogs, networks, sensors, and processors. The connectivity layer provides core communication

and authentication protocols needed for the grid network transactions and also for inter-layer communications between the fabric and resource layers. In addition to providing communication functionalities for the grid, the connectivity layer also provides secured resource-sharing operations for the grid services. The resource layer is located on the top of the connectivity layer and defines protocols similar to the connectivity layer except that they are concerned entirely with individual grid resources instead of the global grid state and the distributed resource collections. Two primary resource layer protocols are the information protocol and the management protocol. The information protocol obtains information regarding the structure and policy attributes of a grid resource. These include the resource configuration, current load, existing usage policies (especially cost of resources). Various user negotiations such as resource requirements, advanced reservation, access priorities, QoS, sharing relationships among users and processes are managed by the management protocols. The focus of the resource and connectivity layers could be summarized as the provision of interactions. The next layer in the grid architecture is the collective layer which provides protocols for global grid interactions. The collective layer contains protocols, services, and Application Programming Interfaces (APIs) that implement interactions across other grids. The top layer of the grid architecture is the application layer. The application layer includes various applications as portals and development toolkits to support the applications. In this layer, users actually connect to the grid via user interface and interact with the rest of the grid. The following sections provide a description of grid middleware and projects that support pricing services.

2.2.1 Globus

The Globus toolkit defines the basic core services required for a computational grid. While constructed as a layered architecture, it sits on the essential low-level core services. Foster and Kesselman [63] describe the Globus Metacomputing Toolkit (or simply Globus Toolkit (GT)) as "a system that specifies the requirements necessary to construct a computational grid" The GT provides a uniformity of platform that orchestrates all computing resources, communication (reliability becomes a concern as individual systems have different architectures and communicate over non-uniform network connections), and security of the computing environment. GT ([64] and [65]) provides a basic Grid middleware based on open standards and components. These components[1] include:

1. The Hypertext Transfer Protocol (HTTP)-based Globus Toolkit Resource Allocation Manager (GRAM) protocol. The GRAM is used for allocating, monitoring and the control of computational resources,

2. Resource reservation and allocation via Globus Advanced Reservation and Allocation (GARA),

3. Authentication and security related services using the Grid Security Infrastructure (GSI), and

4. Distributed access to structure and state information based on the Lightweight Directory Access Protocol (LDAP).

[1]The precise set of components depends on the Globus version.

The current version of the GT is the Globus Toolkit version 4.0 (GT4). GT4 is used to manage, share, and use computational resources across corporate, institutional, and geographic boundaries without sacrificing local autonomy. The GT can be used to build further services and high-level grid frameworks such as Gridbus [66]. In addition to the basic functionality provided by GT4. The GT4 also provides third party solutions. One example is the Swedish Grid Accounting System (SGAS) [67]. The SGAS provides accounting functionality using the SGAS add-on. However, the SGAS lacks economic-aware components such as a proper billing system with pricing, charging, and accounting for commercial services. Other advantages of the GT4 include the use of established standards from communities such as Internet Engineering Task Force (IETF), World Wide Web Consortium (W3C), Organization for the Advancement of Structured Information Standards (OASIS), and Global Grid Forum (GGF). Table 2.1 shows some GT protocols and the standards they use.

Table 2.1: Globus Toolkit Protocols and Standards

Protocols	**Standard**
Secure Socket Layer/Transport Layer Security v1.0 (SSL/TLS)	OpenSSL and IETF
Lightweight Directory Access Protocol v3.0 (LDAP)	IETF
X.509 Proxy Certificates	IETF
Simple Object Access Protocol (SOAP)	W3C
Hypertext Transfer Protocol (HTTP)	W3C
GridFTP v1.0	GGF
Open Grid Services Infrastructure v1.0 (OGSI)	GGF

2.2.2 Gridbus

The Gridbus project [68] is focused on producing a set of economic grid middleware services to support e-science and e-business applications using the computational grid service architecture. One of the components of the Gridbus middleware is the Grid Bank. The grid bank provides the infrastructure for Grid accounting and payment [69]. It uses SOAP over Globus Toolkit's sockets. The grid bank service lacks support for pricing schemes that support reservations or dynamic prices. Moreover, computational resources can have only a single dimension (e.g. CPU cycles).

2.2.3 Condor

Condor [70] [71] is a high-throughput computing system that exploits the idle cycles in a network of workstations. A good example of such a community is the set of personal workstations that sit on the desks of workers in an office or home building. Such a community of workstations is characterized by the possibly heterogeneous nature of each workstation specialized to its owner's needs. Each workstation is likely to have different resources (a different amount of memory and disk size, different range of processor speed, and different software and operating system). These differences in workstation configurations affect the QoS provided in the computing environment. The Condor project addresses concerns in its design principles. Similar to other middlewares, the Condor provides different levels of access rights, computing privileges, and job scheduling.

Legion [72] and Oceano [73] are two other systems that have features similar to those of a grid system. Oceano provides a highly scalable and manageable infrastruc-

ture for operating a grid. In addition, it provides its users with the necessary resources to balance a contracted service level. Oceano attempts to resolve the problems associated with non-dedicated resources for each customer in a grid. These problems include over-resource provisioning and lack of rapid response associated with inflexibility in high cost of providing QoS in a grid system. Oceano provides a cost alternative solution to the QoS problems. It automatically and dynamically reassigns resources that meet resource demands of the customers as the resource become available. In these research efforts, very little attention has been paid on pricing grid resources.

2.2.4 Nimrod-G

Nimrod-G performs the function of a resource scheduler, and also manages task-farming applications [74], [75]. The four components of Nimrod-G include the task-farming engine, a scheduler, a dispatcher, and resource agents. The task-farm engine is a user-defined scheduler system with customized applications. The dispatcher uses Globus for deploying Nimrod-G agents on remote resources to manage the execution of scheduled and assigned jobs. The scheduler has the capability for supporting resource discovery selection, scheduling, and the execution of applications on remote ends. For example, a user could provide parameters such as set deadlines time when processing needs to be completed, and the Nimrod-G does the job of finding the best resources available in the grid that meets the user's deadline in a minimized cost of execution.

Nimrod-G supports applications for user-defined deadline with specific emphasis on budget constraints for managing the supply and demand or grid resources. It

achieves this set objective using a set of trading services called Grid Architecture for Computational Economy (GRACE) [75]. The four scheduling algorithms in Nimrod-G [76] include: (*a*) Cost optimization – uses cheapest resources to meet deadlines at a minimized computational cost (*b*) Time optimization – combines all available resources to process jobs in parallel, (*c*) Cost-time optimization – similar to (*b*), but applies time optimization strategy for multiple resource requests, and (*d*) Conservative time strategy – similar to (*c*) guarantees a minimum budget-per-job for each unprocessed job.

2.2.5 Enabling Grid for EsciencE (EGEE)

Enabling Grid for EsciencE [77] (EGEE) project focuses on maintaining the gLite (pronounced *"gee-lite"*) middleware and on operating a large computing infrastructure for the benefit of a vast and diverse research community. The gLite middleware provides a basic framework for building grid application that leverage distributed computing and storage resources across the Internet [78]. The Distributed Grid Accounting System [79] (DGAS) is an important component of EGEE and it implements a pricing scheme using a Price Authority (PA). The PA assigns prices to the subset of grid resources within its administrative domain [79]. The prices are stored in a historic database and are then assigned manually. One of the shortcomings of this pricing scheme is that the default dynamic pricing procedure increases the price for jobs with lower waiting times.

2.3 Grid Services

Generally, the grid encapsulates modular complexities which are managed by a set of middleware frameworks to meet the diverse users' needs in the areas of infrastructure, collaborations, tailored applications, and domain portals. These are collectively called a middleware. Some of the middleware that are most significant to date include Globus [62] and Nimrod-G [80]. The focus of these middlewares is essentially resource scheduling and management. As a new and evolving area, grid computing has attracted considerable research interest. The most notable of these include the NASA's Information Power Grid [81] which runs on the Globus toolkit and the new grid LCG [40] being constructed for analyzing data from the Large Hadron Collider (LHC) project at CERN [82].

2.3.1 Simple Storage Service (S3)

Currently, there is no cost for utilizing grid resources (for research purpose only, for example [83]). However, the grid is slowing developing into a commercial system due to a large interest in grids for public computing. Hence, a sudden explosion of grid usage is expected in the near future which may lead to . To avoid bottlenecks, Amazon has introduced a Simple Storage Service (S3) [17] for grid consumers. S3 offers a pay-as-you-go online storage, and as such, it also provides an alternative to in-house mass storage. Palankar et al. [17] reviewed the features of Amazon S3, focusing on the core concepts: the security model and data access protocols. After characterizing science storage grids in terms of data usage characteristics and storage requirements, they proceed to benchmark S3 with respect to data durability, data

availability, access performance, and file download via BitTorrent (in order to reduce cost). With this information as a baseline, they evaluate S3's cost, performance, and security functionality.

Palankar et al. concluded by observing that many science grid applications do not necessarily need all three most desirable characteristics of S3: high durability, high availability, and fast access. Finally, Palankar et al. noted that S3's current security architecture is inadequate for science collaborations such as DZero[2] in terms of access control, support for delegation and auditing (pricing), and built-in trusts (which were basically a set of assumptions).

2.3.2 Simple Queue Service (SQS)

Amazon's Simple Queue Service (SQS) allows users to create one or more named queues. SQS supports three basic operations. A named message consisting of up to $256K$ of data and $4K$ of metadata can be written into a queue; one or more messages can be read from a queue; and one or more named messages can be deleted from a queue. When a message is read from SQS the reading process specifies a time lock. While the message is locked, no other read request will return the message. The reading process must delete the message before its time lock expires, otherwise another concurrent process may read the message instead.

[2]The DZero (Dϕ) Experiment consists of a worldwide collaboration of scientists conducting research on the fundamental nature of matter. The experiment is located at the world's premier high-energy accelerator, the Tevatron Collider, at the Fermi National Accelerator Laboratory (Fermilab) in Batavia, Illinois, USA. (http://www-d0.fnal.gov/).

2.3.3 Elastic Cloud Compute (EC2)

Amazon charges separately for computer resources consumed and for bandwidth. The pricing model underwent a significant change on June 1, 2007. This section presents S3 and EC2 pricing models. Amazon's pricing for EC2 is $0.10 per hour for each instance, with fractional hours rounded up. Instances must be shut down with the EC2-terminate-instances command. Instances that have crashed and not automatically rebooted continue to acquire charges. Storage for S3 is charged on a flat basis of $0.15 per gigabyte stored per month, with the amount of data stored being calculated twice each day. Amazon S3 uses buckets (similar to file directory) to store files. A few important S3 commands include PUT (`put filename.xyz s3` – to load a file to a bucket (if bucket is s3)), LIST (`ls s3` – list the contents of a bucket), and GET (`get s3` – download the contents of a bucket). Starting June 1, 2007, Amazon has also charged a per-transaction fee of $0.10 for every 1,000 PUT or LIST requests, and $0.10 for every 10,000 GET requests. Use of SQS is charged at $0.10 for every thousand messages sent. Originally pricing for bandwidth for the Amazon Web Services (AWS) was charged on a flat basis of $0.20 per gigabyte transferred in or out of the Amazon network. Under the new pricing model Amazon charges $0.10 per gigabyte sent from the Internet to Amazon. Bandwidth from Amazon is charged at $0.18 per gigabyte transferred for the first 10 TB of data transferred per month, $0.16 per gigabyte for the next 40 TB transferred, and $0.13 for each gigabyte transferred thereafter. There is no charge for moving data between EC2, S3, and SQS.

2.3.4 Grid Resources for Industrial Applications (GRIA)

The Grid Resources for Industrial Applications [84] (GRIA) is a lightweight economic middleware infrastructure that allows businesses to supply and/or procure and federate computational services on a commercial basis. It enables businesses to use the grid in a secure, inter-operable, and flexible manner based on Web Services and SLAs [85]. The GRIA SLA management service accounts for the use of CPU, current activities, disk space, for jobs with a possibility to define customized resource types. The drawback of the charging service offered by GRIA is that the resources in GRIA have a single dimension and prices are statically defined in the associated SLA document. Other GRIA limitations include the absence of dynamic pricing, lack of resource price history, inability to provide an automated negotiation between two different business processes because of the complexities posed by QoS and SLAs [85].

I have presented some examples of existing pricing schemes in Section 2.3. The resource charging concept is usage based. For example, the cost of computation (to use a CPU based on time) is obtained by multiplying the price per compute cycle with the number of compute cycles that the computation used. This type of static pricing scheme does not provide a fair charge for computations that requires a shorter time to complete. Amazon EC2 charges per instance hour. For example, it offers computational power equivalent to a server with 1.7GHz Xeon CPU, 1.75GB RAM, 160GB HDD, and 250Mb/s of network speed at a price of $0.10 per instance-hour [86]. In a similar approach, SUN Grid charge $1.00 per CPU hour. In contrast to Amazon, SUN does not provide any specification for the equivalence of hardware they offer for use. The SUN Grid is similar to paying a high penalty for jobs that

take longer to complete on average. Although that may seem to charge a flat rate of $1.00 per CPU hour, however, users may actually pay for unused cycles if they over-estimate the compute resource requirements for their jobs. The goal of SUN Grid and EGEE (EGEE uses the same charging technique as SUN Grid, known as Price Authority [79]) is to minimize the queue wait time using economic scheduling. The grid is a dynamic resource reservoir and as a result, static charging (usage-based charging, flat-rate charging, and wait time charging) does not capture the essential goals that I address in my research (see Section 1.5).

2.4 Grid Simulators

This thesis employs the use of a simulator. Simulators are useful to mimic the behavior of a real. In this thesis, I use a simulator instead of a real grid middleware because in a real grid middleware, access is restricted by the administrator. The fundamental advantage of simulators is their independence from the execution platform and ability to control the system variables in order. Simulating a mechanism of a one million node distributed system on a single personal computer is rare. This advantage is made possible because the simulator does not run the real distributed system but an abstract model of it. Table 2.2 provides an overview of some existing Grid simulators: Bricks [87], SimGrid [88], GridSim [3] [89], GangSim [90], and OptorSim [91]. Table 2.2 compares the characteristics of other grid simulators and GridSim.

GridSim is similar to SimGrid (SimGrid2 provides similar abstraction through the notion of Agents) in that it is a discrete event simulator. However, compared to SimGrid, GridSim's design considers the existence of several brokers. GridSim man-

ages several abstractions also called entities. These include user, broker, resource, grid information service, input and output. Users in GridSim are characterized by job type (execution time, number of parametric replications, etc.), scheduling optimization strategy, activity rate, time zone, absolute deadline and budget and their associated relaxation parameters. When the brokers receive the tasks submitted by users they carry out their scheduling algorithm. However, because users must compete for the same set of finite resources, brokers have to find tradeoffs (to meet all project deadlines) among user requirements. GridSim describes its resources as number of processors, cost of processing, performance, internal scheduling policy, workload, and time zone.

One of the significant differences between GridSim and other simulators is seen in the management of inputs to the toolkit and outputs. Comparatively, GridSim is a higher-level simulator designed to investigate interactions and interferences among scheduling decisions taken by distributed brokers. For example, at the user code layer, individual resources, machines and grids can be specified. Also, user job attributes such as the job length, file input and output size (bytes), RAM size, and disk size can be specifies.number of

Table 2.2: Grid Simulators

Grid Simulator	**Motivation**	**Principles**	**Implementation**	**Application (Services)**
Bricks [87]	To study and make comparisons between scheduling algorithms and frameworks.	Event driven simulator with replaceable components	Implemented in Java. Scripts used to specify the environments.	Network traffic monitoring/prediction, Performance analysis of scheduling and replication algorithms in data Grids.
SimGrid and SimGrid2 [88]	To study single client multi-server scheduling systems in heterogeneous environments.	Event driven simulation.	Implemented in C.	Compile time scheduling and Runtime scheduling algorithms.
GridSim [3]	Similar to SimGrid. GridSim focuses on grid economy, Producers / Consumers /Brokers, designed for deadline scheduling studies	Discrete event simulator, Brokers receive tasks, find trade offs acceptable for all users (to meet the deadlines)	Implemented in Java on top of SimJava. Entities run in separate threads.	Cost-time optimization algorithms and economy based distributed resource management.
GangSim [90]	To study the behavior of Virtual Organization schedulers as a function of scheduling policy, resource usage policies and workloads	Discrete event simulator.	Derived from Ganglia. Adds tools for workload specification and to generate Grid environments	Synchronous and asynchronous workloads.
OptorSim [91]	To study data replication: Creation and management of data replicas in different geographical locations.	Sites provide computation and/or data storage resources.	Modular architecture, Simulation based on the architecture of the EU Data Grid project	Several replication strategies.

Compared to SimGrid, which is implemented in C, GridSim is implemented in Java on top of an existing discrete event simulation engine (SimJava) which runs entities in separate threads. GridSim is mainly used to study cost-time optimization algorithms used for scheduling task farming applications on heterogeneous grids. A wider application considers grid economy based distributed resource management using deadline and budget constraints.

2.5 Economics of Grid Marketplaces: Scheduling and Reservations

A primary area of focus in grid economics is related to resource allocation and resource management. Several studies in resource allocation and management have only applied market mechanisms to allocate compute resources to job requesting services. For example, Feldman et al. [92] formulate a resource allocation game and study the efficiency and fairness of the Nash equilibria (see for example [93]) that results. Wellman et al. [94] propose an auction mechanism to allocate distributed resources to users.

Several scheduling approaches have been proposed in the literature, including the Tycoon [95] and Condor systems [96]. Most current systems (for example, [56]) assume the resource requirements can be estimated *a priori* by the users. Condor scheduling systems typically perform a matchmaking function, matching jobs to resources based on resource requirements and available resources. However, several empirical studies ([56], [92], and [97]) have found that users provide very inaccu-

rate estimates of resources required and job run times. Providers encourage users to provide good (accurate) estimates (which allows for better planning) by offering incentives for faithful reporting. However, users still do not give good estimates. Using the backfilling technique ([92], [98]) for example, jobs in the queue that have less computing requirements are executed earlier than jobs that have more complicated requirements. This would mean providing incentives to jobs with low runtime requirements. However, jobs may be evicted from the queue if the actual runtime is higher than the estimate, ensuring that users do not deliberately quote low resource and runtime estimates. The prevalence of inaccuracies in the estimates, despite the incentive mechanisms, show a fundamental problem.

Under-estimated job requests or over-estimated computational job requests can significantly undermine the efficiency of the scheduling algorithms. Normally, users will estimate the costs based on the resource requirements and pricing policies of the grid system. The provider estimated utilization costs may be considerably different from users'estimates. This difference between the providers' estimated utilization cost and the users' estimates (projected cost of resource usage) could cause considerable and undesirable regret on the part of both parties. Solutions are needed to address inaccuracies caused by poor estimates of resource requirements so that buyers can better estimate costs, schedulers can better assign jobs to resources, and resource providers can better plan their computational capacity ahead.

Mutz et al. [99] apply a simple form of batched-queue of jobs j_i for $i = 1, 2, \cdots, n$ waiting to be granted resources; where j_i receives service before j_{i+1}. Owner's requirements determine the resources allocated. Mutz et al. model their payment

function on the assumption that users' behaviors imposes some undesirable external constraints on the jobs in the batched-queue. With specific reference to the job value v_i (currency based), and the delay in total turnaround time d expressed as a tolerance factor, Mutz et al. obtained a job priority model using an efficient design mechanism described in [61]. They proposed a penalty function based on the urgency with which a job owner wants his/her job to be executed. Whenever an urgent request for executing a job is made, a penalty is paid by the job owner and the penalty is disbursed as incentives (i.e., compensation) to the jobs that are bumped. The scheme described in [99] is typical of an airline reservation system. For example, the holder of an air ticket may decide to postpone, advance the travel date or cancel the flight after the flight has been confirmed. In any of these cases, the ticket acquires pays a penalty.

My work is novel in applying real options to the grid resources pricing problem and consideration of QoS. Also, in my grid resources pricing model, I introduce a compensation function called a price variant function (pvf) or simply p_f, details of which are explained later. Kang et al. [100] considered one of the major challenges in managing the shared resources of a computational grid. This challenge is providing flexibility of use of grid resources and satisfying the users' QoS. This challenge becomes more prevalent when resources are orchestrated from less reliable desktop PCs, or a user's requirement is biased toward some specific constraints: e.g., a request for 99% resource reliability for 24 hours. Therefore, matching the cost of resource usage based on the forces of demand and supply may yield undesirable pricing results.

Existing pricing mechanisms (market-based economy) [52] have limitations in controlling mismatches between users' QoS and market resource supplies. These mecha-

nisms do not foster the utilization of the under-used, low-quality resources. A highly available job execution service (HA-JES) [100] dynamically and transparently visualizes underlying low-level computational resources to meet imbalance and unpredictable resource usage. HA-JES sits between a user and grid resources and consists of the underlying underutilized low-quality resource. The HA-JES idea is to build a high-quality resource that satisfies user requirements. It achieves this objective by replicating copies of the grid resources and places the redundant resources close to the tasking that needs them. Using the HA-JES may not necessarily guarantee a high and continual profit for the provider since s/he must maintain a consistent high volume of resources to guarantee availability.

Closely related is the work of Tan and Gurd [57], in which they modeled grid resources as a dynamic, distributed, two-sided market. The problem they attempted to solve is the uncooperative habits of grid resource users where concurrent components need to be co-allocated. They applied a Stable Continuous Double Auction (SCDA) scheme, based on the more conventional Continuous Double Auction (CDA). In their results, the SCDA delivered a continuous resource matching, high efficiency and low cost, allied with low price volatility and low bidding complexity, and achieved superiority in terms of performance over the CDA.

2.6 Financial Options

The literature on financial options show diverse methodologies and approaches for pricing options, for example, closed-form solutions such as the Black-Scholes-Merton [101], and Merton [102] model, Monte Carlo simulation methods (see for

example [103] and [104]), lattice techniques (binomial and trinomial (for example [105] and [106])), and other numerical techniques (for example; Fast Fourier Transform [107], Finite Difference [108], Multithreaded Fast Fourier Transform [109], Second Order L_0 Stable Algorithm [110], and Collateralized Debt and Real Options [111]). These approaches focus on financial options.

In a recent study, Goolsbee and Klenow [112] value consumer products (time-intensive goods such as the Internet access) by the time spent using them. Goolsbee and Klenow [112] have not considered users' flexibility in using resources. A real options framework captures the set of assumptions, concepts, and methodologies for creating decision flexibility in a known future. Flexibility in investment decisions (such as the decision to use, postpone usage to later date in the future) is critical because not all of decisions have value in the future. Therefore, uncertainty abounds in the decision to either exercise or not to exercise options. This challenge in the real options has propelled several recent research efforts.

Current literature on real option approaches to valuing projects presents a real options framework in eight categories described in [113]: option to defer, time-to-build option, option to alter, option to expand, option to abandon, option to switch, growth options, and multiple integrating options. There are also efforts reported towards improving the selection and decision methods used in the prediction of the capital that an investment may consume. Carlsson and Fullér [114] apply a hybrid approach to valuing real options. Their method incorporates real options and fuzzy logic and some aspects of probability to account for the uncertainty involved in the valuation of future cash flow estimates. The results of the research in [114] and [113]

have no formal reference to the QoS that characterizes a decision system. Carlsson and Fullér [114] apply fuzzy methods to measure the level of decision uncertainties; however, there is a lack of indication of how accurate the decisions could be. Other notable literature Gupta [115] and Amico [111] evaluated real options and applied it to spot pricing. Similarly, in [116], d'Halluin et al. applied modern financial option pricing methods to a network investment decision problem. d'Halluin et al framework applies estimation techniques to manage the resulting network bandwidth capacity uncertainties. In contrast to [116], I capture uncertainty in my model parameters using fuzzy logic methods. Fuzzy logic has in the inherent property of determining unprecise and vaguely represented variables such as the satisfaction derived by a user measured as QoS. All these research efforts did not consider the development of a price-based grid infrastructure as a means to capture the value of the *gcc*. In a grid system, resources are non-storable and a user who predicts that he/she might need more computing power in the future must pay upfront today to hold the right to exercise the option when he/she needs the computing resources at the stated future date. My research is unique because formulating/transforming the grid resources pricing problem is done using real option concepts and I verify the results with the actual machines usage pattern in the grid.

2.7 Chapter Summary

In this chapter, I presented a review of the related work in the aforementioned areas and examined them in the context of resource pricing. I examined the charging schemes used by one of the early commercial grid service providers – Amazon. I

observed that there is the need to provide a dynamic pricing approach instead of static charging. The chapter also reviewed the economics of grid market places and observed that existing systems often assume that resource requirements could be estimated a priori by users. This assumption is sometimes in correct due to erroneous job specifications. Some studies that attempt to circumvent the problem associated with erroneous job specifications suggest the application of a penalty function as a way to control users' excesses.

Chapter 3

Methodology

In this thesis, the solution methodology consists of four phases of developments. Phase one of the model design is a description of the model architecture and the model design assumptions. A description of the grid trace data and the analysis which lead to the evolution of the pricing model is presented in phase two. Phase three and phase four are presented in Chapter 6. A complete work flow structure of the research methodology showing the relationships of the progressions of research methodology and phases of development is presented in Figure 3.1. Major highlights of Figure 3.1 are seen in the research progress and refinement blocks. The following section presents more details of the model development.

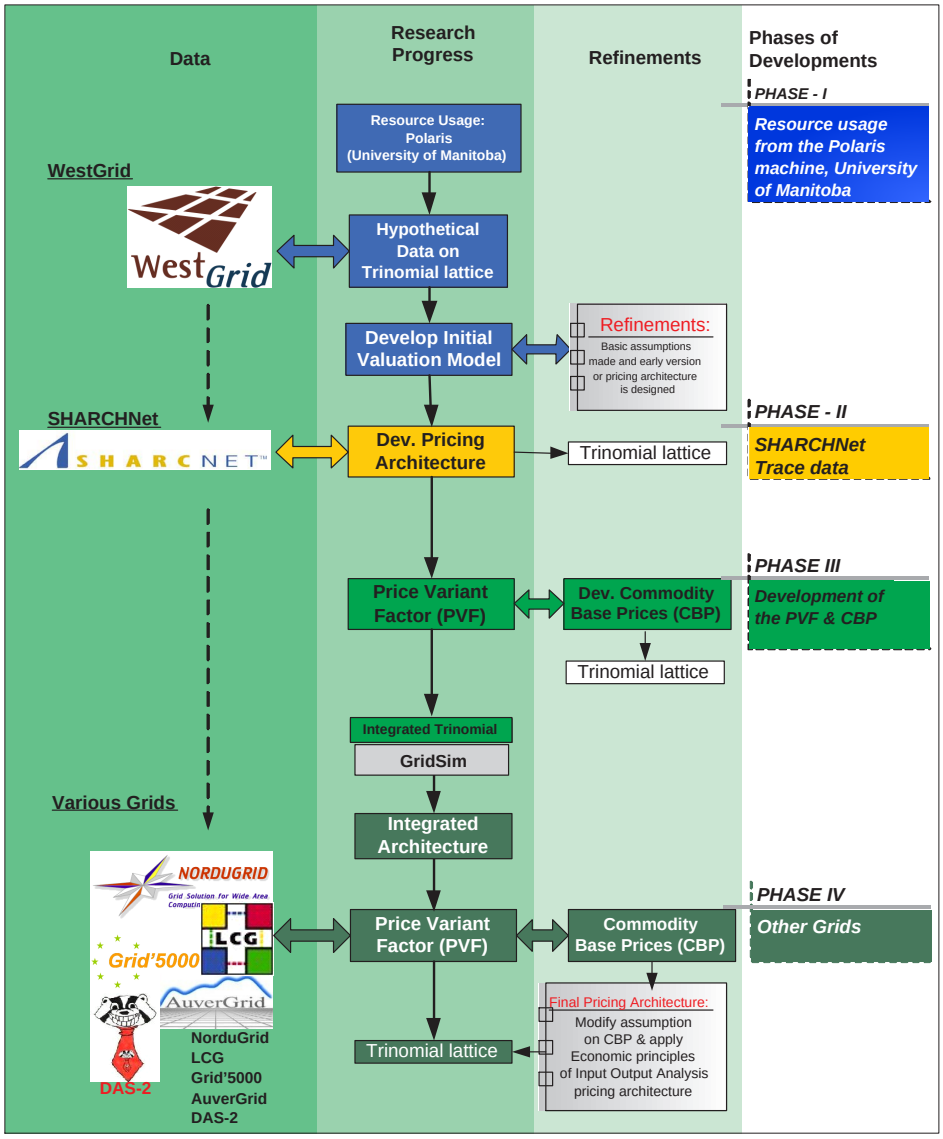

Figure 3.1: Phases of Development.

3.1 Model Architecture and Model Design

3.1.1 Model Architecture

The grid infrastructure is the underlying core for all grid applications. In analogy to the physical infrastructures such as roads, power grids, telephone lines, and

domestic utility systems, the grid infrastructure provides the pieces of distributed computer processes, information, computations, and technologies that are bundled together. These distributed components integrate to form a functional structure. Figure 3.2 shows an abstract representation of the thesis model pricing architecture. This architecture comprises a four-level infrastructure model.

Level-0 – Grid resources: The level-0 of the pricing architecture[1] consists of a pool of available grid compute commodities (gcc) such as CPU cycles, memory, network link bandwidths, disks, various visualization tools, software, and specialized instruments. The grid infrastructure provides a description of the available resources, application capabilities, and defines inter-component relationships among the various clusters that comprise the grid.

Level-1 – Resource modeling: The grid infrastructure provides a description of the available resources, application capabilities, and defines the inter-component relationship among the various clusters that comprise the grid. The grid resources modeling approach facilitates resource discovery, provisioning, and quality of service management.

Level-2 – Monitoring and notification: The function available at level-2 is the provision of updates regarding the states of available resources. At any time during a grid computation, the infrastructure ensures that it provides updates regarding the state of use of resources. These include notifications for changes in projected utilization levels and application notification regarding service

[1]For describing the pricing architecture, the use of "grid" and "pricing architecture" are interchangeable.

changes. The monitoring capability also helps to maintain resource discovery and maintain QoS that is necessary for grid resource usage. In the pricing architecture, resource discovering means ability of the architecture to discover what resources will best be suitable for a specific computation. The monitoring level also provides notifications which may require resource re-deployment or resource re-allocation for higher availability using some form of reservation.

Level-3 – Accounting and auditing[2]: Level-3 of the grid resource pricing architecture consists of user-codes. At this level, several authenticated users log in to the grid to access compute commodities. The objective of a logged-on user is to gain access to the computing commodities for a small price while attaining the highest level of QoS as defined in the SLA. To achieve this objective, the Grid Resources Broker (GRB) or schedulers must map the available physical resources (gcc-s) onto the virtual grid space to make for resource shareability. The accounting and auditing module also uses the generated user and job information in order to transform shared resource usage (utilization) into cost.

This thesis focuses on level-3 of the model architecture – the application layer where a larger part of resource usage is evaluated. Transformation from utilization to cost is achieved when a user's request R_i, for computing resources such as j_i, $(i = 1, 2, \cdots, n)$ is granted. If requests were made in sequence, then R_i receives service before R_{i+1} if all conditions of the Service Level Agreement (SLA)-QoS remain equal. Later in Chapter 4, a financial option based model for pricing grid resources with GridSim is provided.

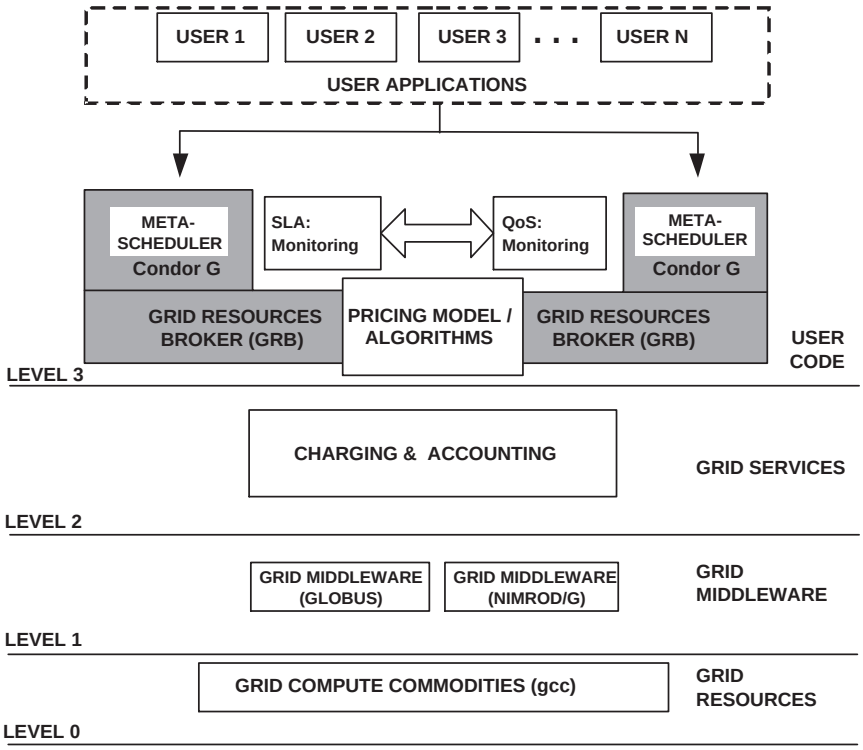

Figure 3.2: Pricing Architecture.

3.1.2 Assumptions and Systems Model Theory

As a preliminary to developing my pricing model for grid resources, and in the absence of grid resource pricing benchmarks I make the following assumptions: (1) I set prices called base prices for the *gcc*-s. These assumed prices are the prices that reflect real market prices but discounted such that the grid provider recovers 100% of the expenses incurred on the infrastructure within a specified period of time without compromising the users' QoS. For example, if $2GB$ of Random Access Memory (RAM) costs \$140, I set a price of \$0.07 a week for 1MB of memory. The option holder has the right to exercise the option any time before the end of the contract (American style option). (2) There are other expenses incurred by the providers, such as cost of

the building, mortgage, equipment insurance, salaries, equipment upgrades, heating, and cooling. In this thesis I do not consider these overhead costs while computing prices for grid resources. This aspect of research has been reserved for future work (Chapter 7). (3) Since the resources exist in non-storable states, I value them as real assets. This assumption qualifies them to fit into the general stream of investment estimates included in the real option valuation approach. A holder of an option has an obligation-free chance of exercising the right to use the grid resources. The obligation-free status enables me to apply existing finance option valuation theory to model my pricing scheme. For example, consider an asset whose price is initially S_0 and an option on the asset whose current price is f. Suppose the option has a lifetime of T. The price can either move up from S_0 to a new level S_0u with a payoff value of f_u, remain steady (without any upward or downward move) at S_0 with a payoff of f_m or move down from S_0 to a new level, S_0d with a payoff value of f_d in these movements. Further discussions on resource pricing using a trinomial lattice approach are presented in Chapter 5.

3.2 Grid Traces and Analysis

One of the challenges faced by grid researchers, grid system administrators, and grid system designers is the lack of availability of grid workload data (trace data). Two important reasons are responsible for the limited accessibility to grid trace data. (a) Security issues and (b) Lack of a standard format to represent the grid trace data. The Grid Workloads Archive (GWA) [31] was created to address this problem by providing a freely accessible website that makes grid workload traces anonymized.

The GWA provides several grid traces using a text-based format. Using packages such as Excel, Statistical Analysis System (SAS), or Statistical Package for Social Sciences (SPSS) analysis can be carried out on the traces. Another workload trace format is the Grid Workload Format (GWF) which is similar to the well-known Standard Workloads Format (SWF) used by the Parallel Workloads Archive [117]. The GWA has an an advantage over the syntectic traces in that workload traces often requires special acquisition techniques which could be reading hardware counters for computer traces, or capturing packets **??**. Obtaining information from several events in the grid is easier because most grid middleware log all job-related events. In addition, the GWA provides a comprehensive summary of real-world grid workload traces. Table 3.1 provides some of the characteristics of the grid traces.

This thesis analyzes the various grid trace data collected from real world grids in four phases[3]. The first stage of phase one trace data analysis verifies the feasibility for the application of a financial option-based pricing approach to price grid resources and the second stage of phase one trace data analysis extracts more detailed characteristics of the utilization trends observed in the collected traces. Based on the utilization, a price variant function – the PVF was designed.

For this research, I collected grid utilization data (grid trace data) from various real grids (experimental grids, production grids, and research or academic grids) and study grid resource users' behavior. The main motivation for this is that there are no benchmarks in grid resource pricing. The results obtained from the trace study and analysis provided a basis for the development of a financial option pricing model to

[3]Note that in this thesis, there are four phases of development. Each phase contains at lease one state activity. Thus, I can have several stages within any phase.

price grid resources as well as a realistic evaluation and justification for the pricing model. I started by collecting data from two Canadian grids, SHARCNet [41] and WestGrid [42], discussed in this chapter, while developing the base price model. Further refinement of the pricing model was done using more grid trace data that includes data from Grid5000 [48], AuverGrid [49], DAS-2 [50], LCG [40], NorduGrid [43] and which are discussed later in Chapter 6 with results and discussions. The following section provides a description of WestGrid and SHARCNet trace data used in phase one and phase two respectively.

3.2.1 SHARCNet

SHARCNet is a institutional (both a research and an academic grid) high performance computing infrastructure and spans 16 leading academic institutions across South Western, Central and Northern Ontario. It is structured as a cluster of clusters consisting of a total of $6,828$ CPUs. Over a period of 1 year, SHARCNet records trace data of about 1.2 million tasks. The purpose for the design of SHARCNet is to meet the computational needs of researchers in a diverse number of research areas and to facilitate the development of leading-edge tools for High Performance Computing (HPC). The SHARCNet trace data that was collected for analysis in this thesis contains about a year's worth of trace records from the SHARCNet clusters installed at several academic institutions in Ontario, Canada. The traces used are for the year 2007, during the period January 01, 2007 till December 31, 2007. The specific resources that are extracted for this thesis include the number of processors, memory, CPU time, various job run times, wait times, and hard disc space required.

Table 3.5 shows an excerpt from the SHARCNet trace. Field descriptions from left to right includes submit time (seconds), wait time (seconds), run time (seconds), number of allocated processors (integer number), average CPU time used (seconds), used memory (KB), requested number of processors (integer number), requested time (seconds), requested memory (KB/processor), user ID, and group ID.

The analysis of the grid trace data was carried out in three phases in order to visualize the progressions of the developments of the grid resource pricing model as previously provided in the phases of model development (Figure 3.1).

3.2.2 WestGrid

In the WestGrid HPC consortium, utilization traces for eight machines (Arcturns, Australis, Bigfoot, Borealis, Cortex, Dendrite, Nexus, and Synapse all located in Alberta) which represents over 43% grid resource utilization by the province [118] were analyzed. Figure 3.5 shows the grid resource utilization by province in Canada. WestGrid operates an HPC, collaboration and visualization infrastructure across Western Canada. The WestGrid compute resources are distributed across fourteen partner institutions in four provinces. Since the University of Manitoba is a partner in the WestGrid consortium, I chose WestGrid as an initial case study for this thesis. Since 2004, WestGrid has built a user community across Canada in disciplines ranging from the sciences and engineering to arts and humanities. Therefore, the collected data are from a wider and a multi-disciplinary user community.

Table 3.1: Field Description of the Grid Trace Data from GWA

Trace Item	**Description**
Submit Time (seconds)	The earliest time the log refers to is zero, and is the submittal time of the first job. The lines in the log are sorted by ascending submittal times.
Wait Time (seconds)	The difference between the job's submit time and the time at which it actually began to run.
Run Time (seconds)	The wall clock time the job was running (difference between end time and start time). Alternatively, "wait time" and "run time" are used as equivalent to "start time" and "end time". Note that when values are rounded to an integral number of seconds (as often happens in logs) a run time of 0 is possible and means the job ran for less than 0.5 seconds.
No. of Allocated Processors (integer)	This is also same as the number of processors the job uses; if the job does not use all of them.
Average CPU Time Used	Both the user and system specific average CPU time is provided in seconds. This is the average over all processors where CPU time is used, and may be less than the wall clock runtime. To derive this average, the total CPU time used by all the processors is divided by the number of allocated processors.
Used Memory (KB)	This provides the average used memory per processor
Requested No. of Processors.	Number of processors
Requested Time	The requested time can either be runtime (measured in wallclock seconds), or average CPU time per processor (also in seconds) – the exact meaning is determined by a header comment. In many logs this field is used for the user runtime estimate (or upper bound) used in backfilling. If a log contains a request for total CPU time, it is divided by the number of requested processors.
Requested Memory (KB per processor)	Requested memory uses a status flag where the flag is 1 if the job was completed, 0 if it failed, and 5 if was canceled.
User ID	An integer, between one and the number of different users.
Group ID	An integer, between one and the number of different groups. Some systems control resource usage by groups rather than by individual users.

3.2.3 Phase One: Feasibility

The objective of phase one of my methodology was to verify the feasibility of the design of a resource pricing model for a computational grid system using financial options. Resource valuation is an important research issue in the implementation of grid resource management. For computational grids, for example, resource management involves (among others), the charging of utilized CPU cycles among various grid users. However, the grid resource users in a grid environment have different computational needs and exhibit dissimilar usage patterns. As a result of the differing utilization patterns exhibited by various users, an understanding of the resource utilization patterns is considered an important step for modeling grid resource pricing. The resource usage patterns enable me to identify users' resource requirements in a grid. Therefore, as a proof of concept and as a case study, I started the resource pricing model design by using the usage activities of the Polaris parallel machine at the University of Manitoba for a period of twelve months (between January 2007 and December 2007). Table 3.2 shows the average monthly CPU utilization obtained from the polaris parallel machine observed during this period. The utilization trend shows that most CPU cycles were under-utilized for the periods of January 2007 to August 2007. For example, in January 2007, an average monthly CPU demand of 135 cycles was requested and only 75 cycles were utilized. It is important to note that CPU cycles is a measure of the numbers of computing power that was required to execute a request in the Polaris machine. In the months of September 2007 to December 2007, the average monthly requests and actual utilization were zero cycles. Using these sample utilization pattern I observe that for optimum gain and justification for

Table 3.2: Average Monthly CPU Utilization in Polaris Parallel Machine in 2007

Month (2007)	**CPU (cycles) Requested**	**CPU (cycles) Utilized**
January	135	75
February	105	95
March	125	80
April	140	80
May	165	95
June	175	100
July	200	100
August	125	100
September	75	75
October	65	65
November	50	50
December	75	75

infrastructure investment in the distributed system the under-utilized cycles should be as close to zero as possible (the case for the months of September 2007 to December 2007). One of the ways to get the optimal benefits of the distributed environment is to utilize the resources to its fullest. In other words, I keep the distributed system busy and eliminate idle cycles. I also noted that within the individual months, availability of the requested cycles tends to fluctuate erratically, making availability a major concern. To capture the behavior exhibited in terms of cycles availability, I treat the cycles as assets and attempt to valuate them using financial options. For this current research (phase one), I observe that the utilization trends do not follow a particular known trend. To verify the suitability of pricing the resources using a financial option model, I introduced artificial spot prices for the cycles at various times of their availability (which is the same as the contract period) as exemplified by the trinomial tree structure of the solution space (further detail is provided in Chapter 5). I then

compute the option values (prices) and study the variation in a space of 6 months to determine the effects of time of exercise on option value. Time of exercise here means the time at which the grid compute commodities are going to be utilized, up to six months in the future. Figure 3.3 shows the effects of time of exercise on the cycles. I run the trinomial lattice using the following hypothetical values: For example, for a one-step trinomial tree I set strike price ($K = \$70.00$), resource price ($S = \$80.00$), expiration time ($T = 0.5$ in years), interest rate ($r = 0.06$), volatility ($\sigma = 0.2$), and the number of time steps ($N_j = 2N + 1$). I extend my study by varying the volatility σ in steps of $0.0, 0.1, \cdots, 0.7$ and $N = 4, 8, 16, 24$ (where N is the step size in the trinomial tree). For a 6 month contract (expiration time in months), for example, $N = 3$ would mean a 2 month step size and $N = 12$ would mean a 2 week step size. At various times (i.e., first month, second month, $\cdots$, sixth month), I have different option values. Figure 3.3 shows option values for three months only. As expected, the option values that occurred at earlier months are lesser than the option value at later months for the same resource. The reason is that a guaranteed availability in terms of QoS is hard to ascertain. The results obtained also shows a dependence on the future price rather than the spot price. I observe that from the date of signing the contract to the actual date of utilization, the price variation is due to various factors such as change in demand on grid resources and advancement in technology. Based on these changed prices of the grid resource commodities (in other words, the underlying assets for the option) the option values are calculated. From the utilization trends in WestGrid the generic problem – that is, to satisfy every resource requested and guarantee their availability to users. Phase one [37] of my analysis also shows that

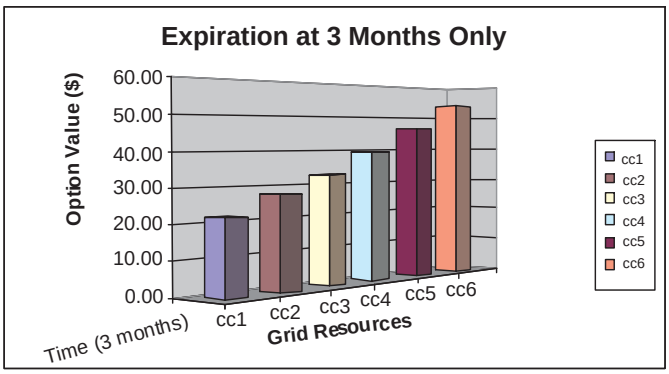

Figure 3.3: Compute Cycles Obtained for Three Months Expiration.

the price of a resource is determined by the effects of time T and volatility σ (one of the most subjective and perhaps the most difficult factors to measure).

3.2.4 Phase Two

Important findings of phase one include a verification of the application of a financial option theory to price grid resource usage. The behavior exemplified in the observed utilization patterns show that users' request for resources are characterized by uncertainties in the decision of the user to make a choice for resource use. To model the associated resource prices, in this thesis I use the application of a real option idea as proposed in [37]. The application of financial option enables the association of the uncertainties present in the observed utilization patterns as exercise opportunities in real options. The principal advantages of using real options include: (a) using real options will help to value the gain of investing now or making follow-up investments later if the original project is successful, (b) real options will also be useful because I am interested in valuing the possibility that a user may decide to opt out from the decision to utilize the proposed portion of the grid compute cycles (this situation

may result in an exit option), (c) real options, when integrated in financial option pricing can help to value the user's inputs and output response to possible changes in technologies (a flexibility option), and (d) generally, real options will enable the valuation of the possibility to wait, learn or deal with uncertainty prior to deciding whether to exercise the options based on the profitability of the exercise.

In the following section, I analyze traces from WestGrid and SHARCNet. The analysis enables me to make specific comparisons of resource usage behavior across various grids. Table 3.3 and Table 3.4 show excerpts of the resource utilization collected from WestGrid (Nexus and Cortex machines) and Table 3.5 shows an excerpt of the resource utilization collected from SHARCNet.

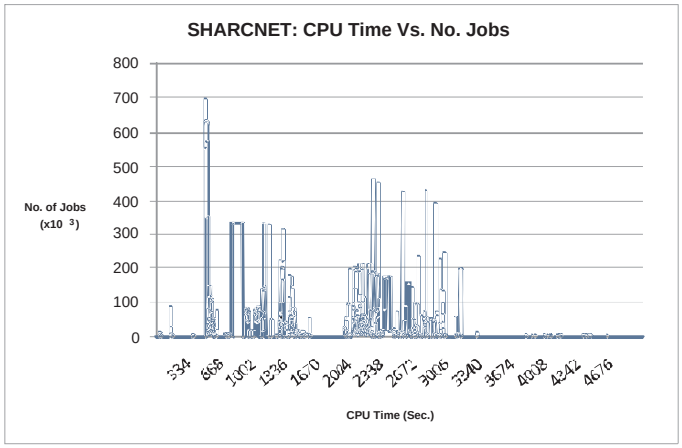

Figure 3.4: CPU Time Vs. Number of Jobs for SHARCNet.

The trace data were collected from January 2007 to December 2007. Characteristics that describe the trace data are provided in Table 3.1. Figure 3.4 shows the used CPU time and the number of jobs it supports in SHARCNet. From this figure, it can be observed that in SHARCNet more number of jobs are supported in the early part of the year. However, there is a sharp drop in the number of jobs supported in the

Table 3.3: Excerpt of Grid Resource Utilization Trace: WestGrid (Nexus Group).

Nexus
1-Jan-2007 - 31 Dec 2007
CPU time correction factor 1

user	group	jobs	cputime	ncpus*wall	ave. cpus	ave. KB memory
chunli	chunli	306	13025:47:47	13278:47:46	2	35123
eblebatt	eblebatt	240	3254:20:20	3360:07:26	1	341790
yko	yko	219	2367:03:34	2392:49:10	1	163755
tmineva	tmineva	17	1268:01:41	1322:59:12	3	97308
pravivk	pravivk	190	860:40:07	901:03:12	1	498339
bsuther	bsuther	90	785:30:31	791:43:06	1	45025
abrantov	abrantov	73	438:45:34	516:27:29	2	202994
mdeo	mdeo	23	660:58:58	457:12:46	2	2869626
ahmedb	ahmedb	30	451:23:16	454:40:46	1	37178
dechaume	dechaume	23	363:30:31	365:33:54	1	578678
dgc2	dgc2	27	255:31:21	256:57:41	1	88680
apenny	apenny	34	236:40:35	249:04:30	3	297912
eec	eec	71	111:51:58	223:01:15	1	35514
mal	mal	34	188:29:44	210:47:33	1	5772

later part of the year. This behavior is somewhat skewed from the grid's expectation: the larger the number of jobs the grid has to satisfy, the higher the CPU time (and the larger wait time) especially if the required computation is resource intensive. An exception is the possibility for most of the jobs to take larger CPU time than necessary. This situation may lead to a waste of compute cycles. It can also be seen that the times of low CPU availability make the grid jobs request more wall clock time. To make the grid busy, I need to attract more users to the grid so that the idle cycles can be turned into productive computations. How to attract more patronage and thereby encourage the grid provider without compromising the service objectives or

Table 3.4: Excerpt of Grid Resource Utilization Trace: WestGrid (Cortex Group).

Cortex
1 Jan 2007 - 31 Dec 2007

CPU time user	correction group	factor jobs	1 cputime	ncpus*wall	ave. cpus	ave. KB memory
kukukova	kukukova	592	10058:03:13	12012:00:3	1	961113
dechaume	dechaume	12	158:26:31	222:44:54	1	727937
ngomez	ngomez	7	100:02:46	102:14:06	1	66543
siegert	siegert	23	23:34:53	61:10:10	1	494694
momid	momid	2	30:01:32	60:16:30	2	24362
keizo	keizo	7	63:27:53	57:00:50	1	3608088
wengler	wengler	2	26:19:52	48:55:52	1	21834
mshaw	mshaw	3	0:00:03	48:01:06	1	1866
jmmbaga	jmmbaga	1	0:19:20	47:25:50	2	23876
zhs064	zhs064	46	23:06:22	39:56:55	1	41320
phillips	phillips	8	23:19:41	25:43:44	1	22582
dsears	dsears	13	0:00:00	25:04:42	1	1033
tmineva	tmineva	1	0:00:00	24:00:34	2	1538
yup03411	yup03411	54	18:04:03	22:32:16	1	12635

quality was a major concern of Phase two. In this thesis, I introduce a control factor called the price variant function (PVF) to modulate the grid resource utilization for optimal profitability for the provider and satisfaction for the user. However, further analysis is necessary to support the conceptual framework for PVF.

3.3 Chapter Summary

In this chapter, I presented my basic design and the approach to develop the model. The objective of phase one and phase two of the design methodology is to test the suitability of applying financial options to price grid resources. In phase one of my

Table 3.5: Excerpt of Grid Resource Utilization Trace: SHARCNet.

```
# Generated by get-clean-logs.py ($Revision: 0.1$) on 2007-05-31 13:15:35.343000
# ------------------------------------------------------------
# System name: SharcNet
# System info: ?
# Sites: 10
# Processors:6828
# CPU Info: ?
# Memory: ?
# Disk space: ?
# Network: ?
# Log source: ?
#
#------------------------------------------------------------
# Format documentation: Grid Workload Format (http://gwa.ewi.tudelft.nl/)
# Field description from left to right:
# (fields contain -1 if not available)
#
```

JobID	Submit Time	Wait Time	Run Time	Nprocs	Ave.CPU	UsedMem	ReqNproc	ReqTime	ReqMem	Status	UserID	GroupID	ExecutableID	QueueID	PartitionID	OrigSiteID	LastRunSiteID	JobStruc	JobStrucParams	UsedNetWk	UsedLocalDisc (MB)	UsedRes	ReqPlatform	ReqNetWk	ReqLocDisc (MB)	ReqRes	VOID	ProjectID
1	1135130133	0	6	1	0.1	-1	-1	-1	-1	1	U1	-1	X1	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	1135130401	1	13	1	0.13	-1	-1	-1	-1	0	U1	-1	X1	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	1135130415	0	17	20	0.01	-1	-1	-1	-1	0	U1	-1	X1	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	1135130438	0	17	1	0.16	-1	-1	-1	-1	0	U1	-1	X2	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	1135130471	0	5	4	0.03	256	-1	-1	-1	1	U1	-1	X3	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	1135130835	0	16	1	0.13	-1	-1	-1	-1	0	U1	-1	X1	-1	1	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	1138120593	0	6	1	0.11	-1	-1	-1	-1	0	U2	-1	X4	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	1138120603	0	5	1	0.1	1024	-1	-1	-1	0	U2	-1	X5	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	1138120664	0	5	1	0.13	1024	-1	-1	-1	0	U2	-1	X6	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	1138120938	0	6	80	0	-1	-1	-1	-1	1	U2	-1	X7	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	1138120952	0	9	80	1.69	51	-1	-1	-1	0	U2	-1	X8	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	1138121042	0	964	80	956.69	14694	-1	-1	-1	0	U2	-1	X8	-1	2	SWF	SWF	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

design, I describe the model architecture and present the design assumptions. In phase two of my design, I described the real grid trace data that I used. I also presented the analysis which evolved from the pricing model and extended trace analysis from WestGrid to traces obtained from SHARCNet. The objective of phase one and phase two of my methodology is a proof of concept for the use of hypothetical data to run trinomial and a design of the pricing architecture. The results of the analysis of phase one and phase two provide lead to the initial version of my pricing architecture which is based on the trinomial lattice approach.

Limitations of the initial pricing architecture include the absence of benchmark values which will serve as my reference point to price the grid resources. This prelimi-

Table 3.6: Grid Resource Usage by Provinces in Western Canada: Source [1]

Province	**Large Shared Memory Usage (%)**	**Message Passing Systems Usage (%)**	**Cluster Usage (%)**	**Average Usage (%)**
Alberta	52.3	44.1	31.4	42.6
British Columbia	17.6	21.0	59.0	32.5
Ontario	14.3	4.3	0.6	6.4
Quebec	13.0	2.5	1.1	5.5
Nova Scotia	2.0	6.7	2.7	3.8
Saskatchewan	0.8	10.1	3.0	4.6
Newfoundland	0.0	6.1	0.3	2.1
Manitoba	0.0	5.2	0.6	1.9
Prince Edward Island	0.0	0.0	1.3	0.4

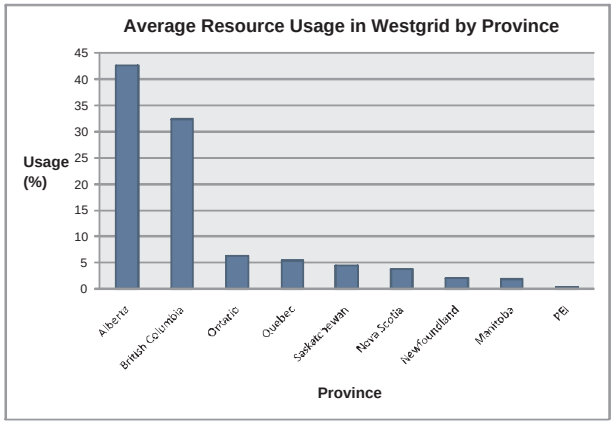

Figure 3.5: Grid Resource Usage by Provinces in Western Canada

nary base model design helped to motivate the need for: (a) a Price Variant Function (PVF) to control prices and increase the utilization of grid resources; and (b) the use of Commodity Base Prices, since there are no standards or benchmarks available for grid resources prices. I will use trace from other non-Canadian grids and in other

application areas. Chapter 6 provides more details of the traces from five other grids.

Chapter 4

System Implementation and Simulation with GridSim

This chapter presents the system implementation and simulation with GridSim. I also describe how I integrate the basic design into GridSim to enable me to apply financial option pricing theory. I also present the user interface design for the simulator which simplified the use of GridSim in my experiments. The user interface was implemented using the Java Abstract Window Toolkit (AWT) that supports Graphical User Interface (GUI) programming.

4.1 GridSim Toolkit Simulator

The GridSim [89] toolkit is based on Simjava2 which is a general purpose discrete-event simulator. GridSim that manages time-variable resource assignments. That is, a user job that runs in GridSim uses variable amount of time which depends on the

job characteristics as specified by the user. Figure 4.1 shows the GridSim multi-layer architecture. The first layer (from the bottom) consists of the Java Virtual Machine (JVM). The first layer manages the events or interaction among GridSim components. The second layer contains the GridSim infrastructure components, such as the resource and network hardware simulator, which are developed using the interfaces provided by the first layer. The third layer is concerned with modeling and simulation of computational grid, and the fourth layer facilitates the modeling and simulation of data grids. Some GridSim components are also extended from the third layer to support the requirements for implementing data grids. These components include grid information service and job management which maintains a log of the grid directory service and maintains a list of jobs to support scheduling respectively. The fifth layer is the simulation of Grid resource brokers or schedulers. The top layer is reserved for customized user code development with different scenarios in GridSim.

To simulate the grid environment using GridSim, grid entities such as resources, users, jobs, and machines are created and assigned during the simulation. The simulation consists of created entities; an example is the gridlet (a *Gridlet* object is a package that contains all information related to a grid job execution, especially, management details such as processing requirements, disk I/O operations, output file size, input file size). In this thesis, Gridlets correspond to the gccs defined earlier. Each gcc is characterized by the required number of Processing Elements (PEs), length of the job (in Millions of Instruction (MI)), and the processing cost required to execute it on the grid. The PEs are then combined to form a machine, where one or several objects of the machine forms a grid resource. A grid resource consists of a set of grid

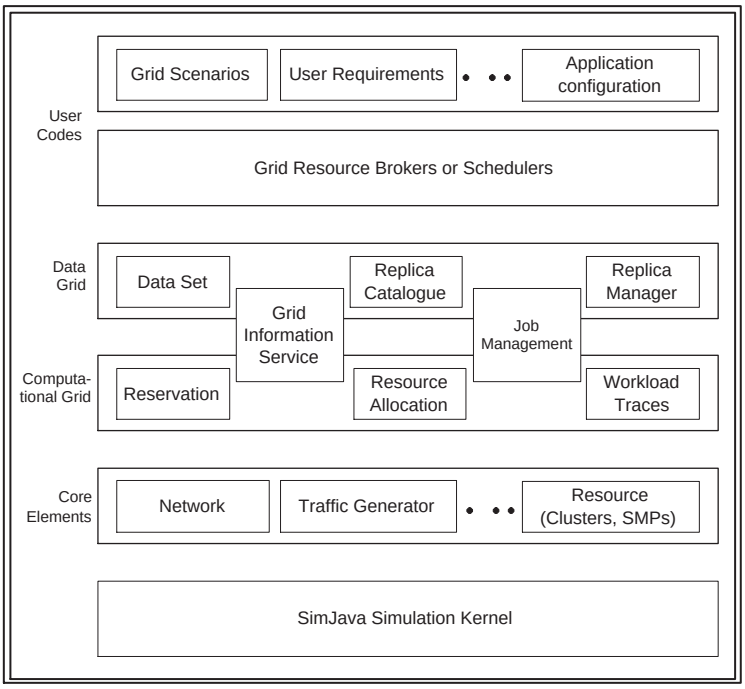

Figure 4.1: GridSim Toolkit Layered Architecture [3].

tasks. Each task (a grid application) uses processor time, computational resources, and some I/O. Figure 4.2 shows *gccs* as resources in a grid scenario. Several PEs form a machine and several machines join together to form the grid. A grid *gcc* can have

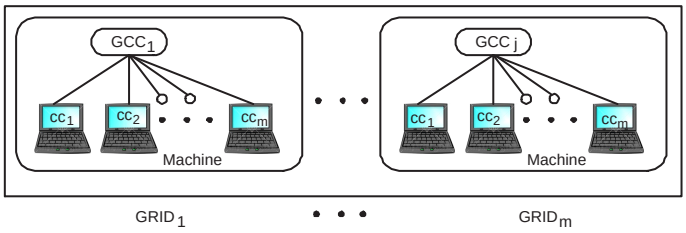

Figure 4.2: *gcc* and Grid Resource Setup.

a minimum of zero resources, however, within a grid the maximum resource available to the user is $j * gcc$ where j is the number of available machines that forms a grid.

4.2 Integrating Pricing Architecture in GridSim

The integrated pricing architecture consists of my model pricing architecture (Section 3.1.1, Figure 3.2) combined with the GridSim layered architecture shown in Figure 4.1. The integration of the pricing model into GridSim is done at the user code level in GridSim. The user code level in GridSim was substituted using the price and usage optimization level in my pricing architecture (Figure 3.2). GridSim uses Dutch auction bidding process to allocate jobs. The Dutch auction was modified to use the reverse (reverse Dutch auction which is described in detail in Section 5.4. Another significant aspect of my model is addition of p_f, SLA and QoS relationships, and generation of a trinomial tree in every round of the bidding process. Chapter 5 explains the trinomial lattice approach for pricing resources. Figure 4.3 shows the resulting integrated simulation-based pricing architecture used in my evaluation.

4.3 Financial Option Pricing and the GridSim Simulation Environment

The simulated grid environment consists of heterogeneous resources and user entities. The computational capacity of the grid is shared among the users based on their computational needs. I consider two computational constraints; the cost of computation and the quantity of time a computation takes to complete. The status information of the resource changes dynamically[1]. GridSim charges the grid users

[1]GridSim manages the dynamic resource changes by regular updates in the Grid Information System module.

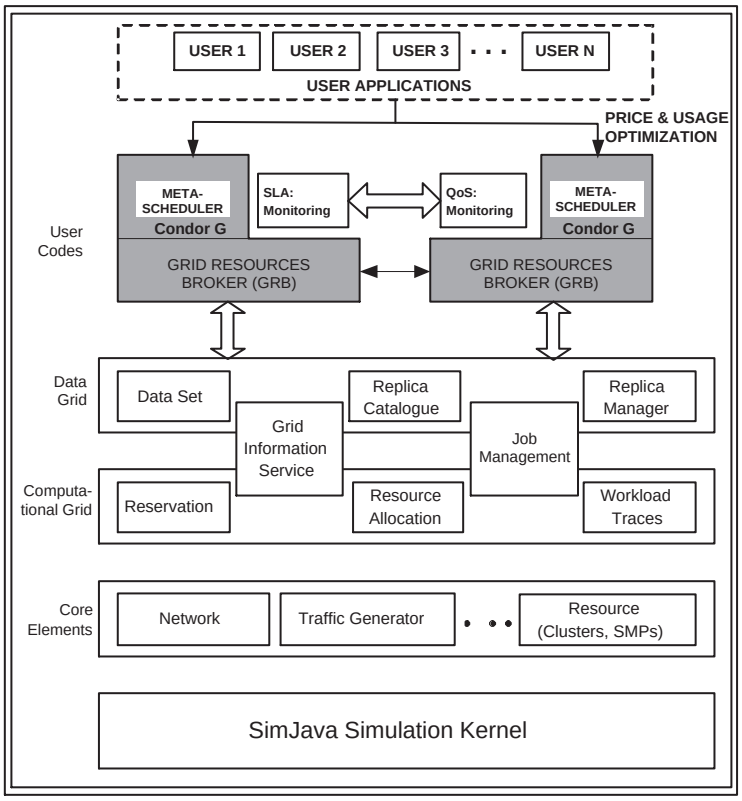

Figure 4.3: Integrated Architecture.

for the portion of the computational resource capacity used. It is rare that these computational resources are allocated totally to a single user; even if a user chooses to purchase an entire resource, I expect at some point that the resource will be partitioned. Although computational resources are not continuously divisible, in my simulation I see all resources (CPU cycles, memory, network link bandwidths, disks, various visualization tools, software, and specialized instruments) as compute commodities which enable me to measure usage quantitatively.

My simulation was carried out on a 32-bit Windows based PC operating system. The system specifications include an Intel processor with a Duo Core 2.20GHz, 4.00

GB RAM, and 320 GB of Hard Disk. Figure 4.4 shows the various stages of the processes carried out in GridSim (GridSim processes). In my simulation using GridSim, I start with stage 1, creation of users and user jobs where the users specify the job characteristics such as expected computation length and file size. Using this information a reversed Dutch auction (details provided in Section 5.4) computes the winners (jobs that wants to use gccs at the computed option value and above) and sets up a priority list in stage 3. At stage 4, I create the various resources which consist of various machines that form a particular grid. Stage 5 of the GridSim processes uses my model which at this stage is already integrated into GridSim in stage 6 to compute the option values for the resource utilization. Figure 4.6 shows the simulation configuration for the experiment. The simulation begins with the creation of the grid simulation scenario. This consists of the creation of PEs, new machines, and resources. The next step in the simulation (after the grid scenario is created) is the creation of the users' scenario which includes the creation of a grid task for each user. The bidding process and integrated trinomial tree are then loaded to start the process of simulation. The configured simulation entities include users, jobs, machines, simulation parameters, job assignments to users, resources, machine assignments, and option pricing settings. The simulation achieves the following; (a) accepts user job requirements, (b) uses reverse Dutch auction to compute jobs that wants to be executed first, i.e. users who are willing to pay anything starting from the computed option value (also seen as the available budget), (c) creates a priority list of the winners based on budget and job requirements, (d) creates grid scenario, (e) price the gccs using the pricing model.

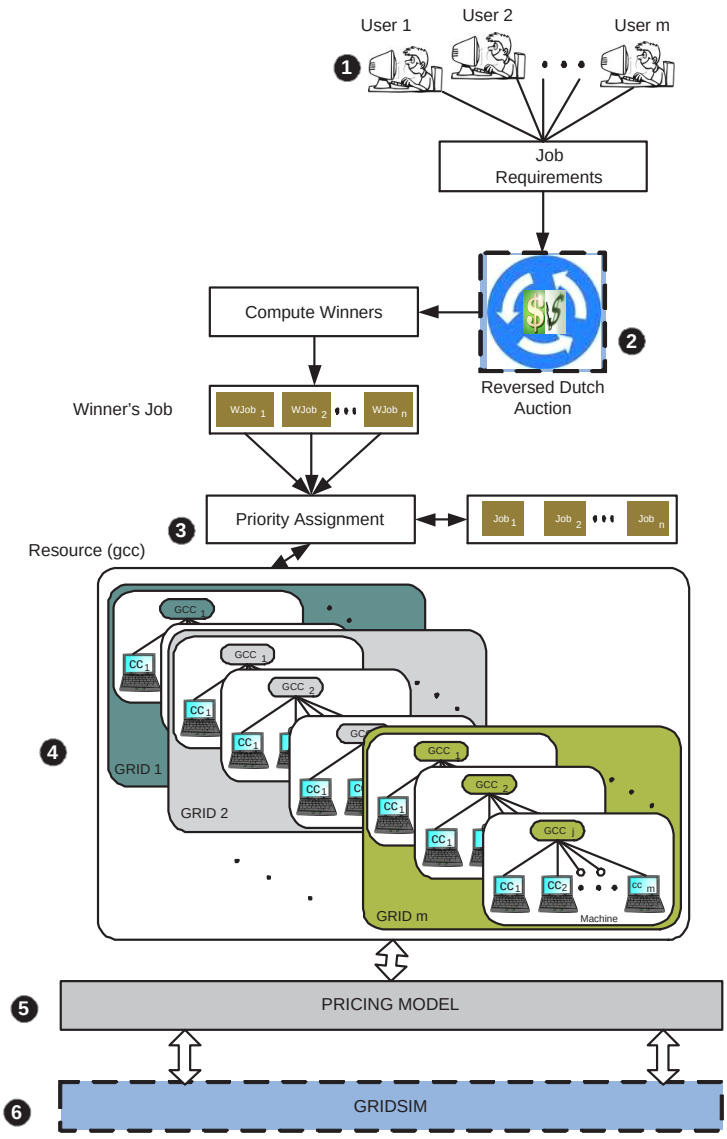

Figure 4.4: GridSim Processes.

4.4 Implementation

In my simulation using GridSim I configure four environments. These include the creation of the grid scenario, creation of individual machines, trinomial tree, and initialization of GridSim to start the simulation. GridSim implements four types of

auctions (first price sealed auction, English auction, Dutch auction, and Continuous Double auction) and their reverses. The Reverse Dutch Auction (RDA) that is native to GridSim applies a method where buyers start the auction and the lowest bid is considered best. In this approach, the Dutch auction becomes ascending and starts with a minimum price going until the maximum price. My pricing model considers users whose willingness to bid for a resource is uncertain at a time as a result of the uncertainty in the resource availability. At this time since the resource availability is uncertain, I use the reverse Dutch auction for trinomial (see Section 5.4).

4.4.1 Simulation Setup

The user interface was implemented using the Java Abstract Window Toolkit (AWT) that supports Graphical User Interface (GUI) programming. The AWT and Java Swing Application Programming Interface (API) enhanced the user interface component as well as flexibility of window layouts. To create the grid scenario I use the GridSim simulator interface. Figure 4.5 shows the user's simulation interface for resource pricing. The Exit option under the file tab terminates the simulator.

4.4.2 Creation of the Grid Scenario

Figure 4.6 shows the simulation configuration interface. Using the simulation configuration interface, inputs to the simulator such as users' jobs, machine configuration, simulation parameters, job assignments, resources, machine assignments, and option pricing parameters are configured prior to the simulation. This interface has a main menu which consists of file, configuration, and simulation functionalities. The con-

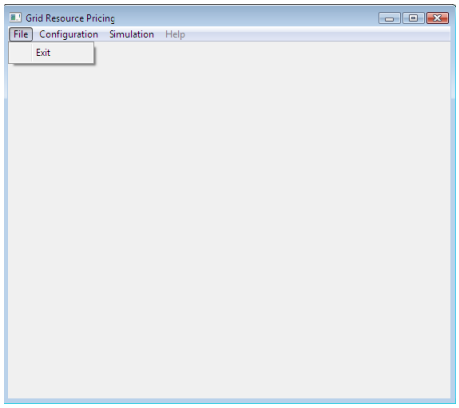

Figure 4.5: Simulation Interface.

figuration tab performs functions which include configuring of users, jobs, machines, simulation parameters, assignment of jobs to users, resources, resources, assignment of machines to resources, and tab to financial option pricing integrated into the GridSim. The simulation sub-menu loads the simulation parameters and starts the simulator. In the following, I provide the setup guide for my implementation.

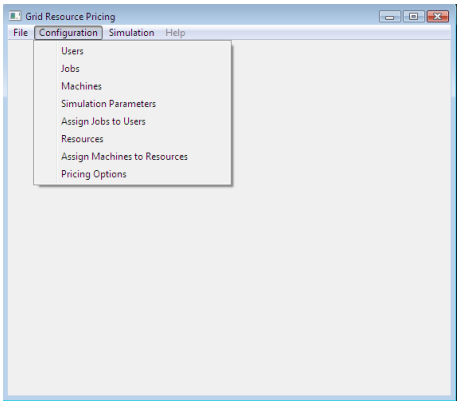

Figure 4.6: Simulation Configuration Interface.

User Configuration Setup

To start the simulation, from the main menu "configuration" enable us to configure or define the grid which include setting up users, jobs, machines, resources, and the pricing parameters. The parameters that define a user include user name, baud rate, budget (the budget defines the willingness to be able to make an early bid in the future), deadline (speculated), and bidding policy applied (where bidding is of type RDA, prioritized cost, prioritized time, or balanced). Figure 4.7 shows the user setup interface for my simulation. At any time before the simulation starts, the configuration file may be edited and any of the inputs modified. At any time the parameters that define the user are completed, one user is set up in the configuration file.

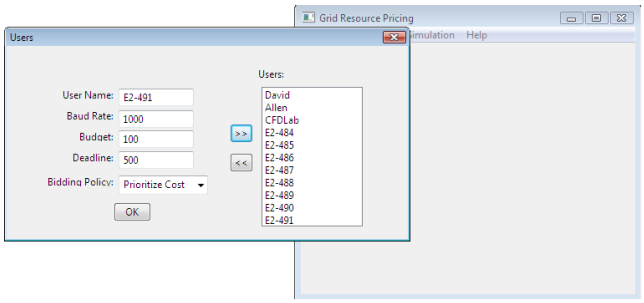

Figure 4.7: User Configuration Setup Interface.

Job Configuration Setup

Figure 4.8 shows the job configuration interface. It specifies the job name, job length (million instruction per second (MIS)), file input size (bytes), file output size (bytes), RAM size (MB), and Disk size (RAM).

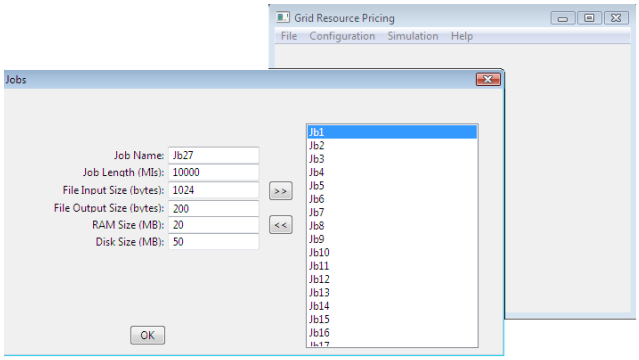

Figure 4.8: Job Assignment Setup.

Job Assignment Setup

The job assignment configuration setup takes a user and maps jobs that are previously defined in job configuration to the user. Figure 4.9 shows the interface for mapping jobs to users.

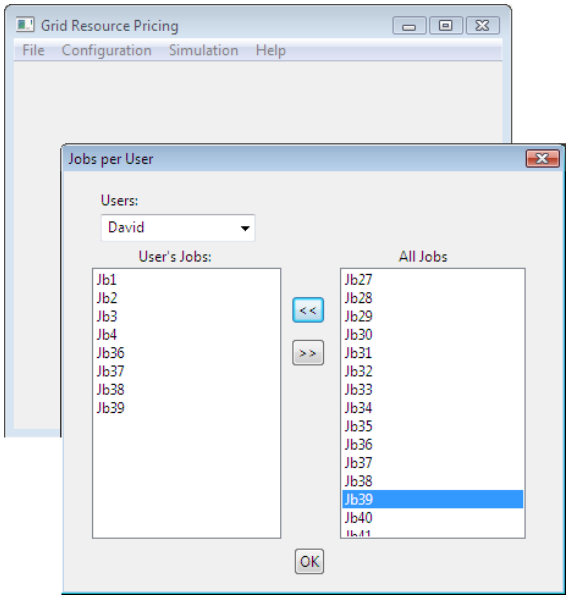

Figure 4.9: User Jobs Assignment Setup.

4.4.3 Creation of Individual Machines: Resource Assignment Setup

Each machine is defined by the following parameters; PEs, million instructions per second (MIPS) per PE, available RAM (MB), and available HD (GB). For every machine specified, a resource is created. Figure 4.10 shows the machine mapping and the resources available in the simulation.

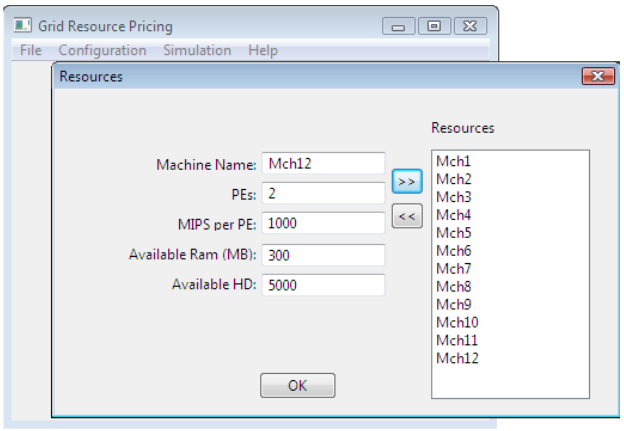

Figure 4.10: Resources Setup.

Assignment of Machines to Grids

Figure 4.11 shows the available machines in the grid (left vertical pane). During the resource assignment setup, machines that were created were each assigned to the grid. For each selected grid, a number of available machines are added.

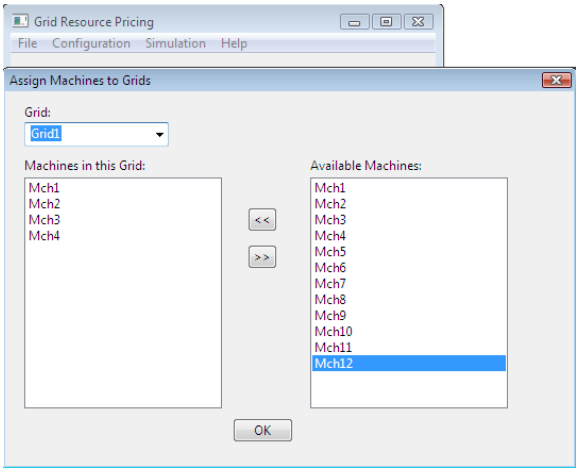

Figure 4.11: Assign Machine to Grids.

4.4.4 Creation of the Grid

Figure 4.12 shows the interface to set up grid specific parameters for my simulation. It specifies the grid name, the base costs, and baud rate. Figure 4.13 shows

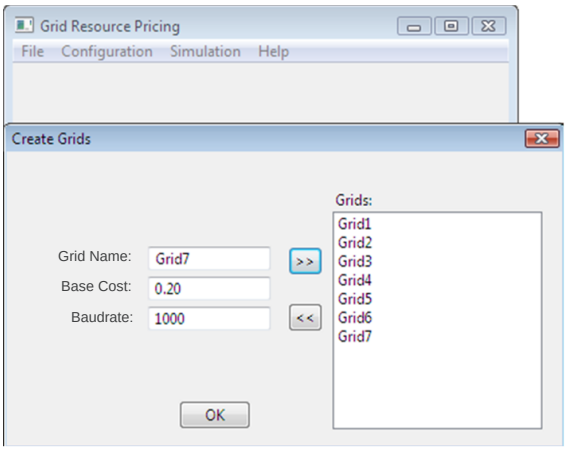

Figure 4.12: Create Grid for Simulation.

the interface to set up grid specific parameters for my simulation. It specifies the grid name, the base costs, and baud rate. Figure 4.13 shows the interface for setting

up option pricing parameters for the trinomial tree. I set up values for the strike price, time in years, commodity base price, number of time steps, volatility, and the incentive opportunity. These parameters are loaded prior to the simulations using the load parameter interface in Figure 4.13 and the simulation starts.

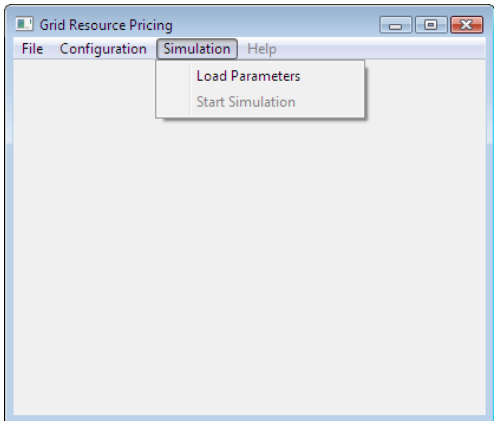

Figure 4.13: Load Parameters to Run Trinomial in Simulation.

4.5 Chapter Summary

In this chapter, I integrated my grid resource pricing model architecture into the GridSim toolkit in the user code layer. I also showed the configurations for the entities involved in my simulations such as users, *gcc*s, and grid.

Chapter 5

Discrete Time Financial Option Model for Pricing Grid Resources

In Chapter 1 I explained that the option price of an asset can be computed with some certainty. However, it is hard to determine the price of a grid resource with any certainty because of the characteristics of the grid resources (Chapter 1) and the non availability of benchmark information to price grid resources. If realistic information about the resource price at expiration becomes available, it can help predict the future grid resource price with near certainty. I use the Cox, Ross, and Rubinstein (CRR) [105] model to build a tree to compute the grid resource price. I begin this chapter with an introduction to the mathematical foundations of my pricing model. The data analyzed in Chapter 3 shows that grid resources utilization evolves unpredictably through time. Since the utilization trend changes in such an unpredictable manner, the resource price follows a stochastic process. I formulate the pricing model in a discrete time framework following an intuitive approach suggested

by the CRR model [105].

5.1 Asset Pricing

I start with a well known hypothesis made by Black and Scholes [101] about asset price behavior. This hypothesis states that future returns on a stock are normally distributed with a mean upward drift, and a standard deviation that can be estimated using historical data [119]. This means that the distribution of the stock price is asymmetric with future values above the current price more likely than values below the current price. In other words, price changes follow a Geometric Brownian Motion (GBM) governed by the stochastic differential equation in Equation (5.14):

$$dS = \mu S dt + \sigma S dz. \tag{5.1}$$

where z is called the Wiener process, drift, μ, and the volatility of the asset, σ, are known constants and dS is the change in the level of the asset price over a small interval of time dt. If I divide all through by S, I have:

$$\frac{1}{S}\frac{dS}{dt} = \mu + \sigma\frac{dz}{dt}. \tag{5.2}$$

In Equation (5.2), the percentage change or the return in asset price $\frac{dS}{S}$, tells me two things; (a) in the time interval dt, the average return on the asset due to drift μ, (μdt) is deterministic (b) added to the drift term is a random component which is characterized by a change dz, in a random variable z and the volatility σ of the asset.

The random variable z is a Wiener process [21]. Following (a) and (b) above, two key properties define the Wiener process; (i) dz is normally distributed with mean zero and variance dt and (ii) the values of dz over two different, non-overlapping increments of time are independent.

5.2 Binomial Tree

The CRR model [105] is the discretized version of Black-Scholes-Merton (BSM) model [101]. I start with the basic CRR [105] model and use it to analyze price fluctuation. The time period T is divided into N smaller intervals: $t_1, t_2, \cdots, t_N$. Suppose S_n is the price at the n-th step, then the CRR model defines the price at the next step S_{n+1} such that:

$$S_{n+1} = \begin{cases} uS_n & \text{with a probability} \quad p \\ dS_n & \text{with a probability} \quad q = 1 - p. \end{cases} \tag{5.3}$$

where u (> 1) and $0 < d < 1$ are the two price movement factors. I compute the option price by building a discrete time and state binomial model of the asset price and then apply the discounted expectations. Figure 5.1 shows the one-step binomial. S_0 is the current asset price (today at the node A). In a given short duration, Δt, the asset price S_0 can either go up with a probability of p_u to a new level uS_0 (of node B) or go down with a probability p_d to a new level dS_0 (of node C) . When the price moves up to uS_0, the pay-off for a call option at node B is f_u ($\max(0, uS_0 - K)$, where K is the strike price of the option) for a call option and when the price moves down to dS_0, the pay-off at node C f_d ($\max(0, dS_0 - K)$, where K is the strike price

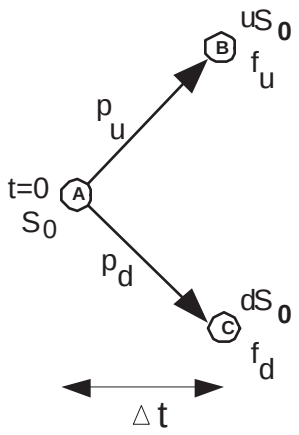

Figure 5.1: One-Step Binomial Tree.

of the option). The BSM model sets up a riskless portfolio that consists of Δ, the number of underlying assets, and an option such that the portfolio value will be the same regardless of whether the asset price goes up or down over the period δt. That is, the option value at the end of the contract term T ($= \delta t$ in one-step binomial tree) for the up movement is $S_0 u\delta - f_u$ and for the downward movement, its given as $S_0 d\delta - f_d$. That is:

$$S_0 u\delta - f_u = S_0 d\delta - f_d. \tag{5.4}$$

or

$$\Delta = \frac{f_u - f_d}{S_0(u - d)}. \tag{5.5}$$

where δ is the ratio of the change in option price to the change in asset price. The variables u and d are the factors by which the asset price moves up or down and are expressed as [105]:

$$u = e^{\sigma \delta t}. \tag{5.6}$$

and

$$d = e^{-\sigma\delta t}. \tag{5.7}$$

The current value of the portfolio for this future option price can be given by the following expression:

$$(S_0\Delta - f_u)e^{r\delta t}. \tag{5.8}$$

where r is the risk-free interest rate. Given that the cost of setting up the portfolio is $(S_0\delta - f)$, the riskless portfolio is:

$$S_0\Delta - f = (S_0\Delta - f_u)e^{-r\Delta t}. \tag{5.9}$$

and substituting in Δ in Equation (5.5) into Equation (5.9), I have:

$$f = e^{-r\Delta t}(pf_u + (1-p)f_d). \tag{5.10}$$

where $p = \frac{e^{r\Delta t - d}}{u-d}$ is called the probability of an up movement in the asset price and $p_{f_u} + (1-p)f_d$ is the weighted sum of the future pay-off and f is the current value of the option. In other words, the value of an option is the discounted value of the weighted sum of the future pay-off. Figure 5.2 shows a two-step binomial tree and the pay-off and option values are given by:

$$f_u = e^{-r\Delta t}(pf_{uu} + (1-p)f_{ud}). \tag{5.11}$$

$$f_d = e^{-r\Delta t}(pf_{ud} + (1-p)f_{dd}). \tag{5.12}$$

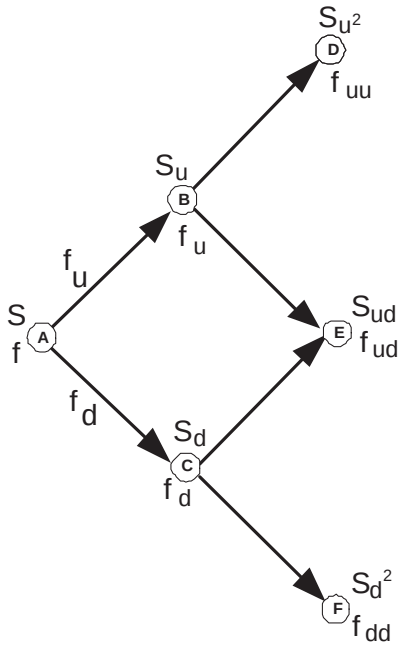

Figure 5.2: Two-Step Binomial Tree.

$$f = e^{-2r\Delta t}(p^2 f_{uu} + 2p(1-p)f_{ud} + (1-p)^2 f_{dd}). \tag{5.13}$$

5.3 Trinomial Tree

In this section, I extend the binomial approach to trinomial tree to a price grid resources. The novelty of the extension from binomial to trinomial in my model is in the application of my compensation function, which, conceptually is similar to an airline system in my model. In an airline system for example, a reverse Dutch auction (Section 5.4) could be used to find someone willing to delay travel plans on an overbooked flight. In this case, the airline applies a strategy of raising the compensation until someone raises his or her hand (a willingness) that ends the auction. A passenger unwilling to give up his or her seat for a free lunch may be more likely to

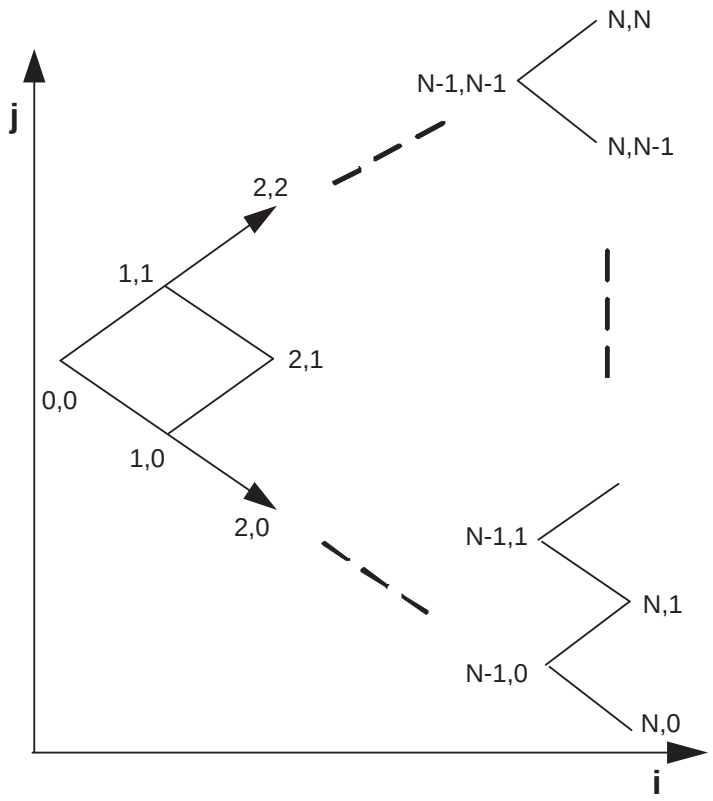

Figure 5.3: Multi-Step Binomial Tree.

relinquish it for free flight, $20.00, or other higher compensatory values (incentives). I model the received incentive as a compensation function which I call the Price Variant Function[1] (PVF), p_f. The primary goal of my model is to recover grid infrastructure expenses within a stated time (two years for example). To achieve this objective, I develop my model in analogy to an option with a dividend paying underlying asset and use the p_f as a control function.

I consider a trinomial tree of asset price in a small time interval Δt based on the original binomial tree in CRR[105]. A trinomial tree has three branches that proceed

[1]I use p_f and PVF interchangeably for my computation notation and readers' convenience.

from each of the tree nodes.[2] Figure 5.4 shows a one-step trinomial tree. From node A in the figure, there are three branches: $x + \Delta x$ at node B with a transition probability p_u, x at node C with a transition probability of p_m, and $x - \Delta x$ at node D with a transition probability of p_d. The transition probabilities are constrained to be non-negative (i.e., $p_u \geq 0, p_m \geq 0, p_d \geq 0$) and the sum of the probabilities is equal to one ($p_u + p_m + p_d = 1$). The price dynamics for a dividend paying asset in the BSM

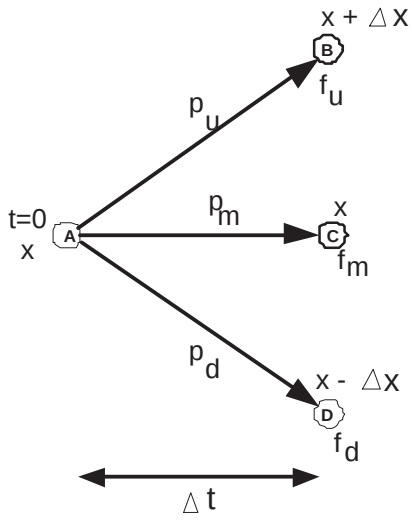

Figure 5.4: One-Step Trinomial Tree.

model is

$$dS = (r - \delta)Sdt + \sigma Sdz. \tag{5.14}$$

The dividend in my grid pricing model is the incentive given to the users of the grid for changing their job schedule, which is controlled through the price variant factor p_f. Therefore, Equation (5.14) reduces the drift rate by an amount p_f and we can

[2]The number of branches could also be more than three and can vary from node to node, in which case, it is called a multi-nomial tree.

re-write the basic Equation (5.14) as:

$$dS = (r - p_f)Sdt + \sigma Sdz. \tag{5.15}$$

Second, similar to the binomial tree, I set the parameters of the asset price which can be captured in the simplified discrete process using $\Delta x, p_u, p_m$, and p_d, where Δx for a trinomial tree is given by $\Delta x = \sigma\sqrt{3\delta t}$ [105]. Following similar steps as for the binomial lattice, the probabilities in a trinomial can be derived as follows:

$$p_u = \frac{1}{2}\left(\frac{(\sigma^2\delta t + v^2\Delta t^2)}{\Delta x^2} + \frac{v\delta t}{\Delta x}\right) \tag{5.16}$$

$$p_m = 1 - \left(\frac{\sigma^2\delta t + v^2\delta t^2}{\Delta x^2}\right) \tag{5.17}$$

$$p_d = \frac{1}{2}\left(\frac{(\sigma^2\delta t + v^2\delta t^2)}{\Delta x^2} - \frac{v\Delta t}{\Delta x}\right) \tag{5.18}$$

Figure 5.5 shows an n-step trinomial tree. For number of time steps (horizontal level), the number of leaves (height) in the tree is $2n + 1$. That is, the number of nodes at any time step i, is given as $2i + 1$. An indexed node has a pair (i, j) where i (row index) points to the time level and j (column index) indicates the distance from the top. The time corresponding to a level i can be calculated $t = i\delta t$. From Figure 5.5, node (i, j) is thus connected to three other nodes (A) in the upward move $(i + 1, j + 1)$, (B) steady move to node $(i + 1, j)$ and, (C) downward move to node $(i + 1, j - 1)$. The option price and the asset price at node (i, j) are denoted as $C[i, j] = C_{i,j}$ and $S[i, j] = S_{i,j}$ respectively. The asset price could be computed from the number of up and down moves required to reach (i, j) from the root node $(0, 0)$

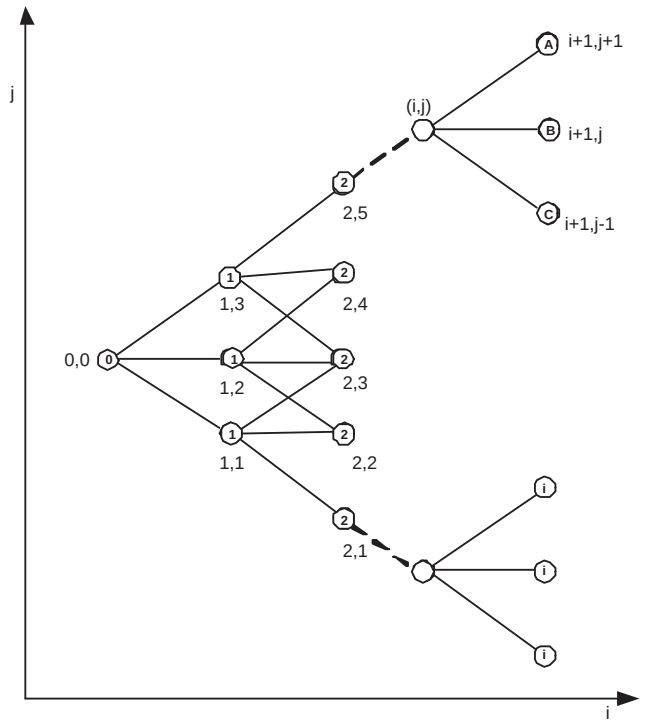

Figure 5.5: n-Step Trinomial Tree.

and is given by $S[i,j] = S[0,0](u^i d^j)$. The option value at maturity is determined by the local pay off. So for a call option, the pay off $C_{n,j} = Max(0, S_{n,j} - K)$ and for a put option, $C_{n,j} = Max(0, K - S_{n,j})$, in both cases where value K represents the strike price. To compute option prices, I apply the discounted expectations under the risk neutral assumption. For an American put option for $i < n$

$$C_{i,j} = max(e^{-r\delta t}(p_u C_{i+1,j} + p_m C_{i+1,j+1} + p_d C_{i+1,j+2}), K - S_{i-j}). \tag{5.19}$$

A grid is a multi-resource system with many grid compute commodities $gcc_i = \{gcc_1, gcc_2, \cdots, gcc_N\}$ where N is the number of available grid compute commodities. For each gcc_i I fit a trinomial lattice with μ_{gcc_i} and σ_{gcc_i}, as the expected growth

rate and volatility respectively. For each such tree (*gcc*) I compute the option value and combine all these option values from various trees according to certain weights attached to each *gcc* (user specified) to find the price on the set of *gccs* requested by the user. Faster processors or latest technology could make a particular *gcc* of higher value than another *gcc*.

5.3.1 Integrating p_f into the Trinomial Algorithm

As mentioned, I developed my model in analogy to options with dividend paying asset and use p_f as a control function. Let the incentive on a grid resource be δ, then the price on the resource becomes $(r - \delta)$, and it is necessary to substitute this price into the formulas. Thus, the probability of an up movement becomes:

$$p = \frac{e^{(r-\delta)\delta t} - d}{u - d} \tag{5.20}$$

At any time $i\delta t$, the nodes on the trinomial tree correspond to the asset prices given as:

$$S^* u^i d^{i-j} + \delta \qquad j = 1, 2, \cdots, i+1. \tag{5.21}$$

when $i\delta t < \tau$ and

$$S^* u^i d^{i-j} - p_f \qquad j = 1, 2, \cdots, i+1. \tag{5.22}$$

when $i\delta t \geq \tau$. τ is the time when the incentive is offered. The *gcc* price will be reduced after this date. If I have incentives, then I must ensure that the risk-neutral probability stays positive at all times.

5.3.2 American Style Option Pricing Algorithm

Algorithm 1 shows trinomial tree algorithm that integrates incentive p_f for pricing grid resources assuming American style options.

Algorithm 1 American Style Option: $(K, S, N, T, \sigma, r, p_f, dx)$

$\delta t = \frac{T}{N}$
{initialize parameters}
$v = p_f - \frac{1}{2} * \sigma^2$
$edx = exp(dx)$
$p_u \leftarrow 0.5 * ((\sigma^2 * \delta t + v^2 * \Delta t^2/\Delta x^2) + (v * \delta t/\Delta x))$
$p_m \leftarrow 1.0 - (\sigma^2 * \delta t + v^2 * \Delta t^2)/\Delta x^2$
$p_d \leftarrow 0.5 * ((\sigma^2 * \delta t + v^2 * \Delta t^2/\Delta x^2) - (v * \delta t/\Delta x))$
$disc \leftarrow exp(-r\Delta t)$
{Calculate asset prices at maturity}
$St[-N] = S * exp(-N * dx)$
for $j = 1$ to $2i + 1$ **do**
    $St[j] = St[j - 1] * edx$
end for
{Calculate option values at maturity}
for $i = 1$ to $i + 1$ **do**
    for $j = 1$ to $2i + 1$ **do**
        $C[i, j] = max(0, K - St[j])$
    end for
end for
{Step back through the lattice}
for $i = 1$ to $i + 1$ **do**
    for $j = 1$ to $2i + 1$ **do**
        $C[i, j] = disc * (p_u * C[i + 1, j + 2] + p_m * C[i + 1, j + 1] + p_d * C[i + 1, j])$
    end for
end for
American.put= $C[0, 0]$

5.4 Reverse Dutch Auction (RDA)

In a classical forward Dutch auction (for one item only), the auctioneer begins with a high price and then continues to lower the price until a successful bidder wins the bid. For multiple items, the auctioneer continues to reduce the price until either all items are sold or the seller's reserve price is attained. For simplicity of presentation I assume one user and one resource system in my computational grid.

In a Reverse Dutch Auction (RDA) operation, I assume that the resource users have a level of willingness w_i initially set to 0, i.e., $w_0 = 0$ to pay a price p_0 for utilizing the resources. At this stage, the number of bids submitted N_b is also set to 0 (initially $N_b = 0$). Following these initializations, I allow an increment in the willingness by resource users to increase from the initial w_0 to w_1 for the same p_0. At this stage, I let the price corresponding to w_0 (that is p_0) become the reserve price of the resources. The bid for resources with a user's favorite prices continues while I check for availability of more bids. If there are any bids (increase the number of bids until there are no more bidders), allocation of resources is based on the total capacity, GCC, available after committing some gcc resources based on the willingness w_i on these gccs, where $GCC = \{gcc_1, gcc_2, \cdots, gcc_n\}$. That is, the available resource can be seen as:

$$GCC = \sum_i w_i * gcc_i \tag{5.23}$$

where $gcc_i \in GCC$. Allocation of the resource is computed and the winner gets the allocation; and for remaining capacity and willingness to pay (this time willingness is increased from w_1 to w_2 and the new reserve price of the resource becomes w_1) is

updated. The process continues by checking for more bids until the max grid capacity is attained, that is, during any N_b, the total capacity requested should not exceed the available capacity, $\sum_i gcc_i * w_i \leq GCC$.

5.5 Chapter Summary

In this chapter, the mathematical foundation of my computation using trinomial-based option pricing was presented for resource pricing in a grid. The idea of this chapter is based on the trace analysis carried out in Chapter 3. The novelty of the trinomial algorithm is seen in the introduction of the p_f function. The consequences of my integrated pricing function were also investigated in a sample run. Without a p_f in the trinomial tree, for a hypothetical set of parameter conditions ($K = \$100, T = 1, S = \$100, \sigma = 0.2, r = 0.06, \delta = 0.03, N = 3, \Delta x = 0.02$) I computed the option value as \$10.30. Using these same hypothetical values, but, introducing a p_f of 0.03, where the value of p_f was chosen based on the remaining GCC in the grid system I a computed an option value of \$2.05. That is my new trinomial framework will attract more users wanting to patronize the grid since I obtained a reduction in resource price from \$10.30 to \$2.05. This means that using the p_f-based resource pricing could attract more users to the grid and will help the providers in recovering the infrastructure cost. This chapter also introduced a reversed Dutch auction using the willingness of users as a strategic means to win bidding prices.

Chapter 6

Results and Discussion

In Chapter 3, I described the initial financial option based pricing model using the trace analysis (phase one and phase two) of two Canadian grids (WestGrid and SHARCNet). In Chapter 5, I refined the initial model by introducing an incentive-paying opportunity in the option value computation using a price variant function, p_f. However, in the absence of existing standard for pricing grid resources, I carry out further trace data analysis of four real grids, two experimental grids (Grid5000 and LCG) and two production grids (NorduGrid and AuverGrid) across several application areas. To price the grid resources using financial options, I refined the model presented in Chapter 5 in the following ways; (a) I expanded the trace analysis to other grids (phase three and phase four), (b) I integrate Commodity Base Prices (CBP) into the pricing model, and (c) I experiment with the trinomial lattice approach in the integrated architecture (trinomial-GridSim) to compute option values.

This chapter discusses the findings of this thesis in terms of trace data analysis, conceptual ideas such as Commodity Base Prices (CBP), price variant function, and

experimental results using the integrated GridSim simulator.

6.1 Grid5000 and NorduGrid

Grid5000 is a 5000 CPU nation-wide infrastructure that supports research in grid computing [120]. Grid5000 is an experimental grid platform with nine sites geographically distributed across France. Each of the distributed sites has one or several clusters to a total of fifteen clusters. The traces obtained from Grid5000 were recorded by the batch schedulers that handle the individual clusters. The collected traces include periods from June 2008 to June 2009.

Grid trace data were also collected from NorduGrid [121]. In Scandinavia and Finland, NorduGrid operates as a production grid. NorduGrid runs a testbed based on Globus toolkit. Since 2003 NorduGrid has been used for a production grid computations. It offers light weight general purpose and portable grid middleware. Another advantage of the testbed is that it requires minimal changes to adapt it to experiments and it has a large pool of CPUs (approximately 1000) which are accessible 24 hours a day and 7 days a week. In NorduGrid, non-dedicated resources are connected using the Advanced Resource Connector (ARC) as grid middleware. The status of all the sites composing the NorduGrid is available through the online ARC Grid Monitor [121].

6.1.1 Phase Three: (a) Grid5000

The resource usage patterns obtained from Grid5000 and NorduGrid include memory usage, average processor utilization, and number of processors. Using the traces collected I show the relationship between the number of jobs executed in the Grid5000 and the CPU time needed to support the jobs. Figure 6.1 shows utilization over time for the research grid Grid5000. For ease of comparison with other real grid traces used in this section, the number of jobs executed in the grid is depicted on the vertical axis while the utilization time (in seconds) is scaled on the horizontal. Figure 6.1 also depicts that a larger number of jobs are supported by the grid at CPU time ≤ 10000 seconds. At CPU times greater than 10000 seconds, the number of jobs supported by the grid is negligible. I interpret this behavior to mean that only a small number of users actually submit very long running jobs to the grid or only a few of the jobs submitted to the grid are actually supported by the grid. The consequence of a low utilization of the grid resources mean that it will take a much longer time for the provider to recover the investment on infrastructure. To make a quick recovery on investment on the infrastructure, the provider may want to charge resource utilization at a higher rate or apply a static or fixed cost for resource utilization (as in the case of Amazon). What the p_f does is to control the cost of using resources in a way to attract more users at a reduced rate.

Another perspective of the utilization as seen in Figure 6.1 is the possibility that some jobs may have occupied larger CPU time than was actually specified. When fewer jobs occupy larger CPU time, it results in under-utilization of the resources by the processor. In Grid5000 (in grid generally), without any loss of generality, I assume

that most jobs are sent to a grid because they are resource intensive. Therefore, it is implied that these times of low CPU availability (causing the jobs to stay longer) are due to a wait state (where priority jobs are served by the grid).

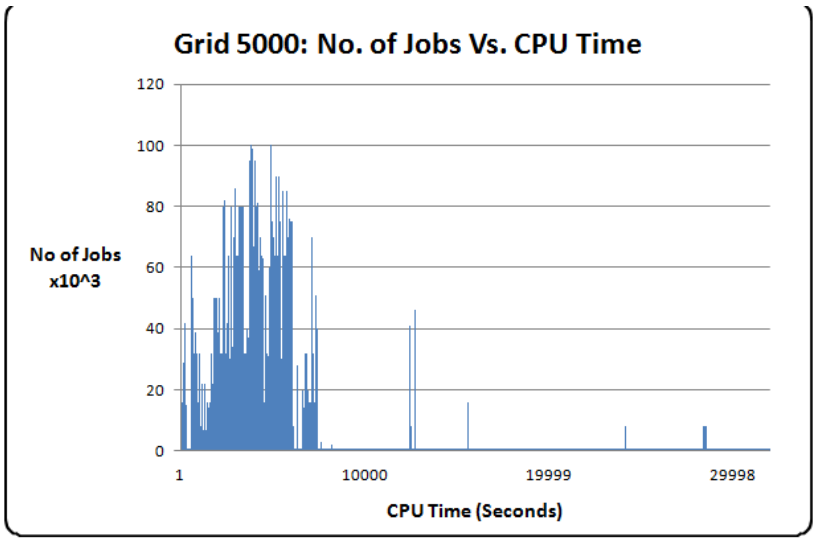

Figure 6.1: Job in Grid5000 by CPU Time (Utilization Trace Collected June 2008).

6.1.2 Phase Three: (b) NorduGrid

Figure 6.2 shows the job utilization in NorduGrid with a peak value of 10×10^3 jobs compared to Grid5000 which has a spike value of 100×10^3 jobs. NorduGrid has low utilization on average (10% lower than Grid5000). Comparing the jobs supported in Grid5000 and NorduGrid, it is observed that NorduGrid has a somewhat ideal expectation of resource utilization, in the sense that at any time, the grid has jobs running. The NorduGrid, therefore, meets one of the set goals that describes optimal resource utilization. However, the average number of jobs that run on NorduGrid is less than the average number of jobs that run on Grid5000. With a low ratio of

number of jobs of NorduGrid to Grid5000 (1 : 10), more CPU time is used up in NorduGrid to execute a small number of jobs. For this scenario, it can be inferred that the cost recovery on infrastructure investment may take a longer time to attain. Expediting the cost recovery time on infrastructure investment is one of the major goals of my model design.

The characteristics of the NorduGrid depicted in the trace analysis in Figure 6.2 reveals that more years will be required to recover the investment on infrastructure compared to my set limit of 2 years. The drawback of a prolonged delay in recovering of the grid infrastructure expenditure (beyond 2 years set in this study) is that newer technologies may emerge to replace existing ones. When this happens, it becomes hard to attract users that prefer newer technologies to use the grid resources, and older infrastructure becomes obsolete and underutilized.

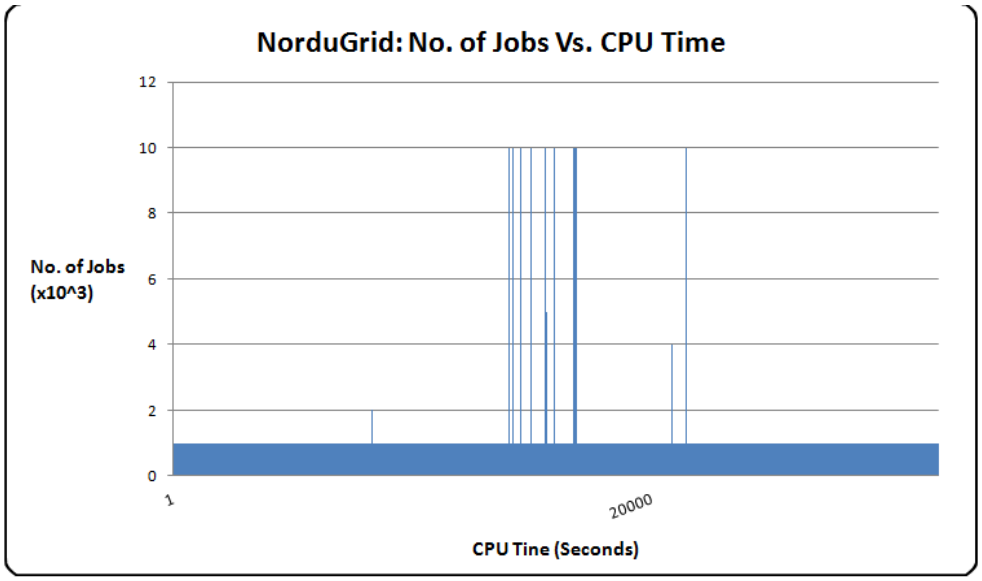

Figure 6.2: Job in NorduGrid by CPU Time (Utilization Trace Collected June 2008).

6.2 Price Variant Function Design

The p_f is a fuzzy controller. Its value ($0 \leq \mu(p_f) \leq 1$) expressed as a fuzzy membership function, depends on changes in technological developments such as new and faster algorithms, faster and cheaper processors, and changes in access rights and policies. Whenever such improvements in technology happen, users' behavior also changes in response to the observed technological changes. One of the obvious effects of faster algorithms, faster and cheaper processors is that more users will be willing to use these to complete their computations in a shorter time. Hence, more users have to be accommodated to avoid idle resources. To meet the goal of shorter recovery time for the cost of the grid infrastructure, I allow the membership function of p_f to vary between 0.0 and 1.0 in a range of choice of categories that allows the users a wide window of opportunities. The function of the p_f is to identify changes in the technology etc., the computed option value ($C_{i,j}$ in Equation (5.19)) into the class of flexibility opportunity (real option) using the trapezoidal membership function as shown in Figure 6.3. Equation (6.1) presents the trapezoidal membership function of p_f:

$$\text{trap}(t : p_{f_a}, p_{f_b}, p_{f_c}, p_{f_d}) = \begin{cases} 0 & t < p_{f_a}, \text{and } t \geq p_{f_d} \\ 1 & p_{f_b} \leq t < p_{f_c} \\ \frac{t - p_{f_a}}{p_{f_b} - p_{f_a}} & p_{f_a} \leq t < p_{f_b} \\ \frac{p_{f_d} - t}{p_{f_d} - p_{f_c}} & p_{f_c} < t \leq p_{f_d} \end{cases} \tag{6.1}$$

where ($t < T$) represents a small window of opportunity available to the user. The user can either exercise (the options), wait (while expecting a better choice in the future), select an alternative (to take advantage of other available choices at a trade-

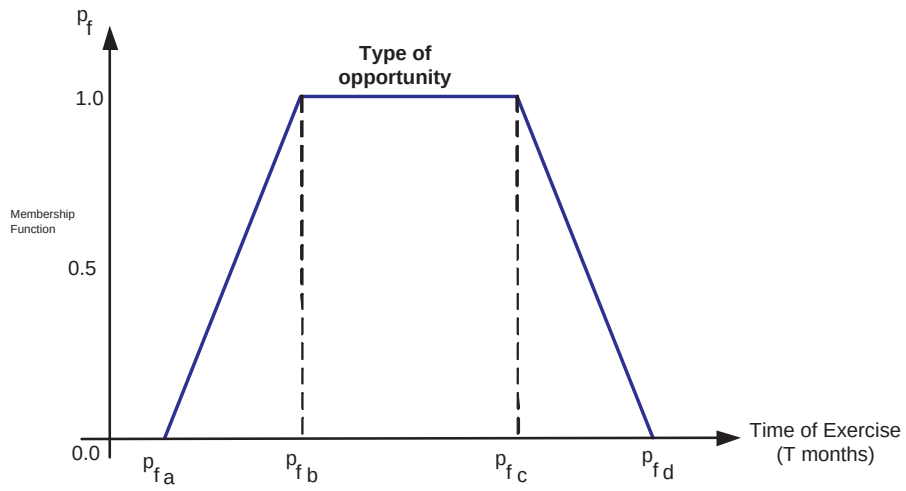

Figure 6.3: Price Variant Function (p_f).

off that is not considered a disadvantage to the user), defer (the right of exercise to a known future date), or abandon (an option that will never be exercised). The individual opportunities in Figure 6.4 such as exercise, defer, alternative, wait, and abandon can be mapped to the fuzzy membership function of p_f in Figure 6.3. In order words, a classification of the opportunities available to the user mapped into the fuzzy membership function such that for each class of user opportunities, and for each of the levels in the corresponding membership function, there is a corresponding QoS also classified with respect to the user opportunities as very low QoS (vLQoS), Low QoS (LQoS), Mid QoS (MQoS), high QoS (HQoS), and very high QoS (vHQoS).

6.3 Price Variant Function

Figure 6.5 shows my design of the p_f controller. Using fuzzy logic to capture the inherent uncertainty to classify the Quality of Service (QoS) in terms of the price of

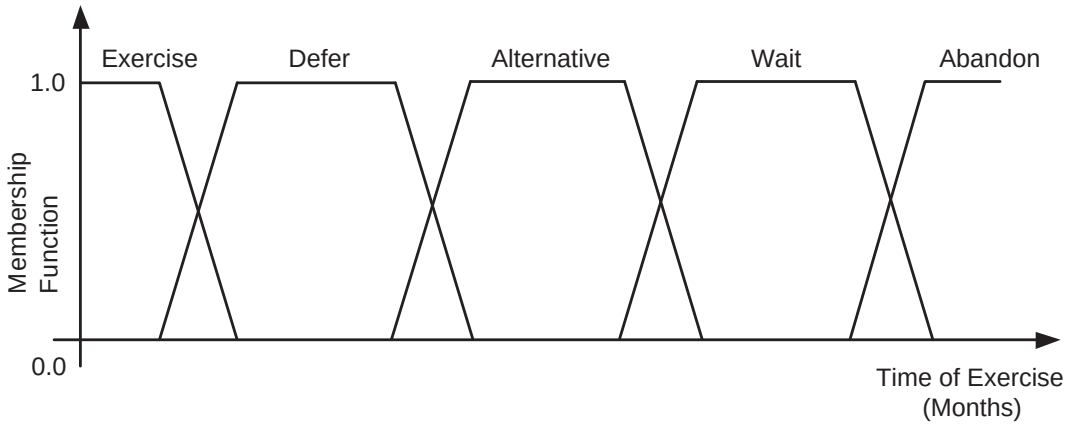

Figure 6.4: Trapezoidal Membership Function for Flexibility Opportunity.

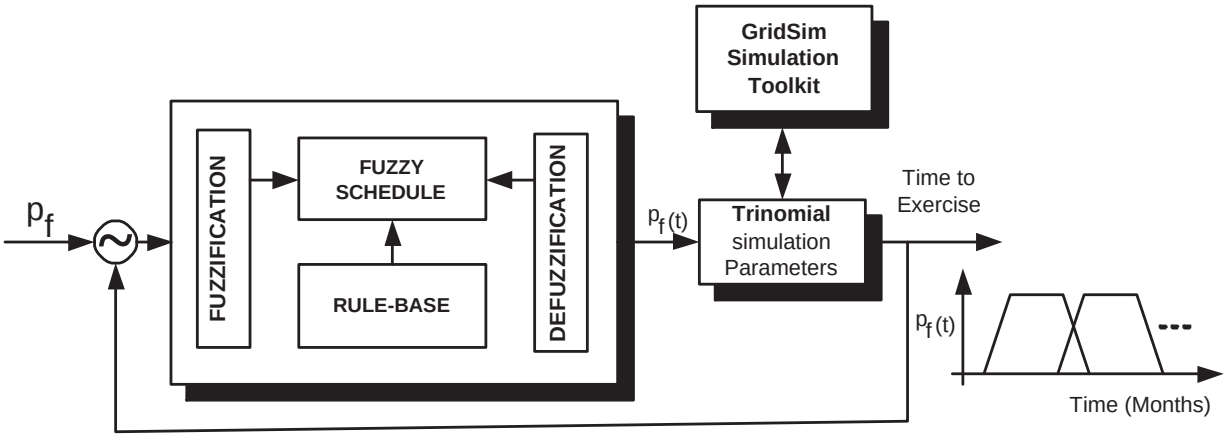

Figure 6.5: Price Variant Function Controller.

a resource and the service quality received. The classification follows the class of real option defined as a trapezoidal membership function as in Figure 6.4. The certainty in predicting the effects caused by technological developments could be captured by integrating p_f and the trinomial approach in Algorithm 1. For example, if the grid resources are under utilized (as in the case of NorduGrid in Figure 6.2 and in Grid5000 in Figure 6.1), p_f could be set to a value that yields a positive dividend $p_f = r - \delta$.

6.4 Commodity Base Price (CBP)

Grid computing (though a relatively new phenomenon), has recently become a popular means for executing computationally-intensive and resource-intensive applications. The nature and vastly differing characteristics of the grid make it difficult to price the resources using a few available economic models. There are no pricing standards available in the literature since (a) efforts are focused more on one aspect of the grid such as resource management (allocation, scheduling, and security) and the development of standards for middleware and grid infrastructure and (b) the use of grid resources has been available for free (for various government funded projects and in academic research) and as a result, research efforts have neglected the need to price the resources. While considerable research efforts and time have been devoted to (a) and (b) above, there are no standards for pricing grid resources. In the absence of a pricing standard, this thesis develops a base resource price to start with for pricing grid resources for utility purposes. A standard Commodity Base Price (CBP) is required for use across various grids. The CBP is also required to help fix a time (for example, 2 years) to recover 100% of the infrastructure cost for the grid provider. The resulting minimal prices will attract a larger number of users to keep the grid busy.

To test the relevance of CBP in my model, I configure the parameters as follows: Asset Price S, Strike Price K, Time to Maturity T, Interest Rate r, Number of Steps N, Interval time between 2 steps Δt, and Volatility σ. I show first results for 3 grids, G_1, G_2, and G_3; 3 users, user_1, user_2, and user_3, and 15 grid resources. I let the users provide specifications for their jobs and apply the inverse Dutch auction to allocate

the jobs based on the users' needs. Figure 6.6 shows my first test case to verify CBP. In G_1, I set a CBP at a presumed value of \$2.00, (i.e., G_1 : \$2.00), G_2 : \$2.50, and G_3 : \$3.00. In Figure 6.6 G_1 executes 5×10^3 jobs at a CBP of \$2.00, G_2 executes 2×10^3 jobs at a CBP of \$2.50, and G_3 executes 1×10^3 jobs at a CBP of \$3.00. In Figure 6.7, I reversed the CBP values for the three grids shown in Figure 6.6.

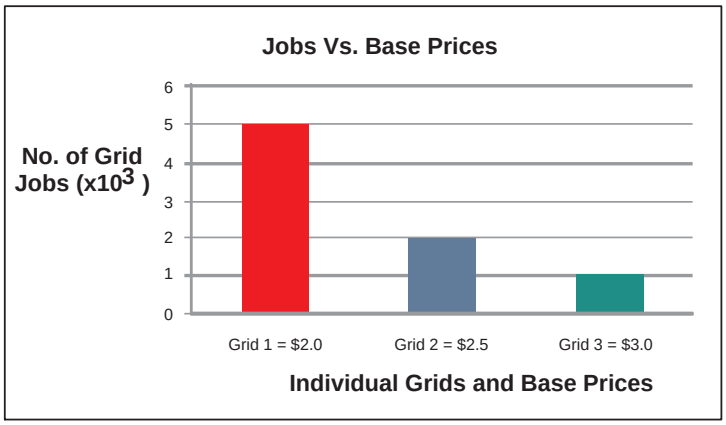

Figure 6.6: Increasing Commodity Base Prices (G_1 : \$2.00, G_2 : \$2.50, G_3 : \$3.00).

The result shows that only 1×10^3 grid jobs are executed in grid G_1 compared to the 5×10^3 in Figure 6.6. Following the results and observations in Figure 6.6 and Figure 6.7, I set base prices of the individual resources based on the current market value as followings:

- For a 2GB of RAM that costs \$140.00, I set a base price of $\$9.589 \times 10^{-5}$/day/MB
- A 200GB Hard Drive (HD) disk whose market price is \$100.00, I set a base price of $\$6.849 \times 10^{-5}$/day/MB
- Processor cycles are charged at $\$6.849 \times 10^{-5}$/day/MHz of CPU cycles.

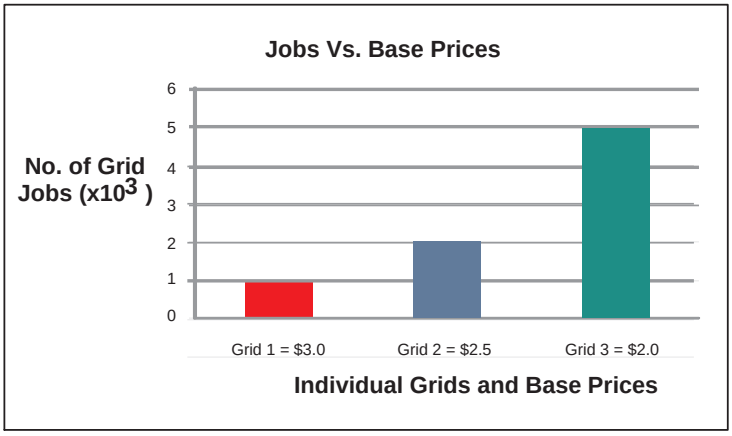

Figure 6.7: Decreasing Commodity Base Prices ($G_1 : \$3.00, G_2 : \$2.50, G_3 : \$2.00$).

The justification for the base prices that I set is based on the static charges that existing commercial grid operatives offer as well as the market value of the resources. For example, in Section 1.1, I provided a summary of the static charges for the use of grid resources in the United States of America and in Europe in Table 1.1 and Table 1.2 respectively. Table 1.3 shows the summary of the commercial industry, minimum cost/hour, cost/month as follows: Amazon, \$0.08/hour, \$537.60/month; GoGrid, \$0.04/hour, \$255.35/month; FlexiScale, \$0.53/hour, \$358.50/month; Mosso, \$0.10/hour, \$645.12/month; ElasticHosts, \$0.76/hour, \$510.76/month; and Joyent, \$5.95/hour, $\$4,000.00$/month. My base prices of $\$9.589 \times 10^{-5}$/day/MB for the memory, $\$6.849 \times 10^{-5}$/day/MB for the HD, and $\$6.849 \times 10^{-5}$/day/MHz for the CPU cycles are insignificant compared to any of the commercial grid resource providers.

6.5 Phase Four: AuverGrid and LCG

Figure 6.8 shows the actual resource utilization trace in AuverGrid. The trace data of the AuverGrid were collected at the same time of the Grid5000 and the NorduGrid. The figure shows that as AuverGrid supports more jobs with an increasing CPU time as the number of jobs increased from from 0 to 100×10^3. Although, the grid provided support in terms of CPU time, the number of jobs running in the grid experienced a sharp drop to 5×10^3. Similarly, Figure 6.9 shows the number of jobs supported in LCG and the corresponding CPU time. In Figure 6.9, increasing the CPU time does not necessarily mean executing more jobs in the grid. For example, the CPU time increased from 10000 seconds to 19999, and the corresponding number of jobs remained low on average. Comparatively, the LCG supports more jobs at similar CPU times as compared to AuverGrid, however, the idle times of LCG are higher than the idle times of the AuverGrid. The depletion in the number of jobs in LCG means a longer time to recover the investment on infrastructure.

6.6 Experiments and Results

In my experiments, I price grid resources by simulating my trinomial-based GridSim (GridSim with an embedded trinomial lattice – (see the integrated pricing architecture in Figure 4.3)). I run the trinomial lattice using the following model parameters: $S = \$6.849.00 \times 10^{-7}$, $T = 0.5$, $r = 0.06$, $N = 4, 8, 16, 24$, $\sigma = 0.2$, and $N_j = 2N + 1$; I vary $K = 6.847.00 \times 10^{-7}$ for both advantageous (in-the-money options), $S > K$ for call (user) or $S < K$ for put (provider) and disadvantageous

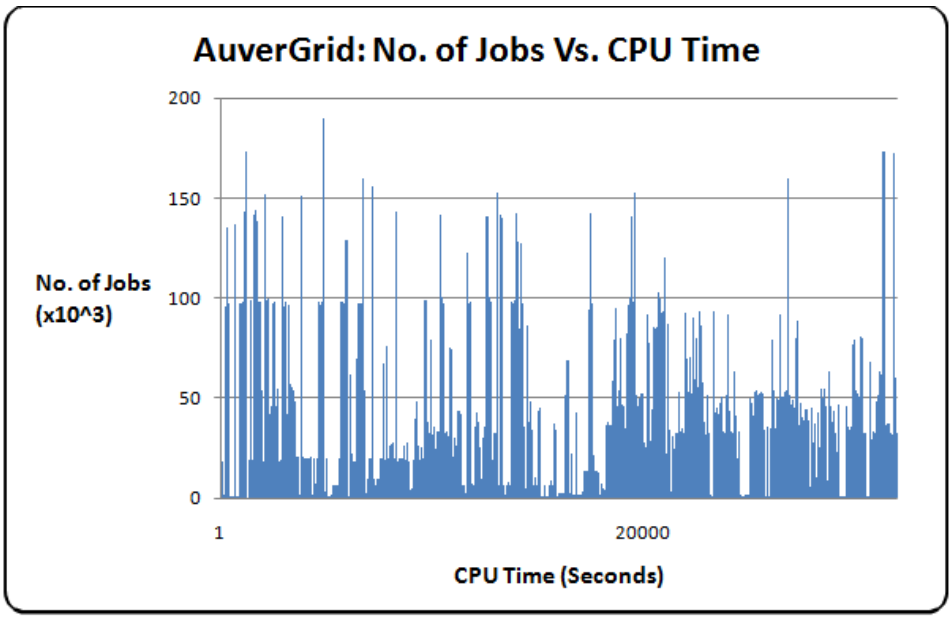

Figure 6.8: Job in AuverGrid by CPU Time (Utilization Trace Collected June 2008).

(out-of-the-money) $S < K$ for call (user) or $S > K$ for put (provider) direction for the users and providers. For a 6 months contract, for example, $N = 3$ would mean a 2 months step size and $N = 12$ would mean a 2 week step size. For a user in-the-money means the contract price is lower than the market price and out-of-the-money means the contract price is higher than the market price. On the other hand, for a resource provider, in-the-money means the contract price is higher than the market price and out-of-the-money means contract price is lower than the market price. My discussions in this chapter are mainly from the users' perspective.

For a call option, I simulate the effects of time of use of one of the *gcc*-s such as memory (RAM), hard disk (HD), and CPU. In other words, I study the effect of exercise time of the option. I start with memory (one of the *gcc*-s) using the parametric values (provided above). The value of K was varied for both advantageous

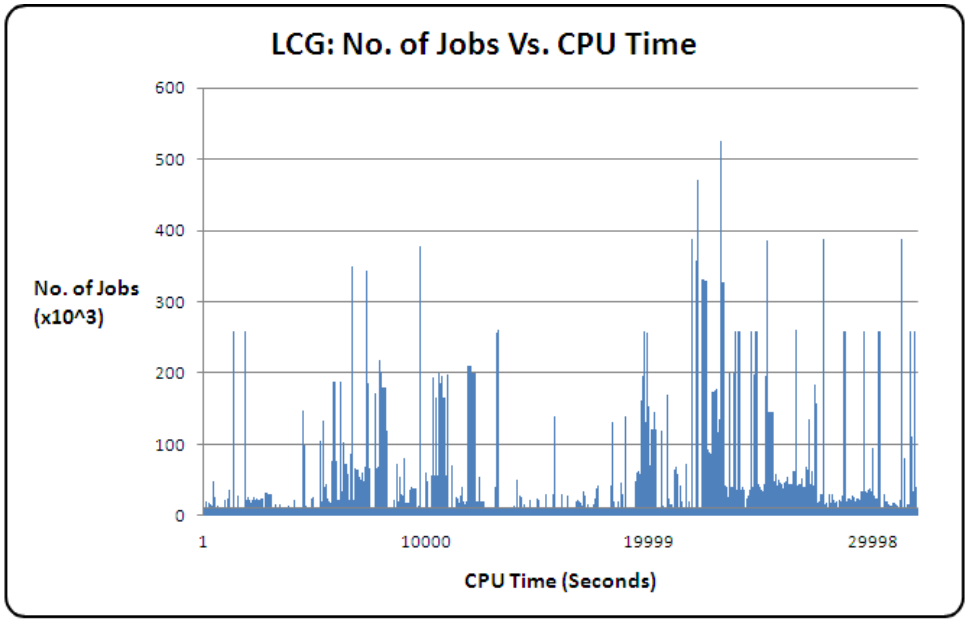

Figure 6.9: Job in LCG by CPU Time (Utilization Trace Collected June 2008).

(in-the-money) and disadvantageous (out-of-the-money) options for the users. These values reflect the market value of this raw infrastructure which I obtained from my commodity base price mapping from the traces to the raw infrastructure available in example grids. This is true for other *gcc*-s such as CPU and hard disk. In the simulation, I compute option values for several step sizes. I analyze the effects of the variations (uncertainty) that exist between the total period of the option contract and the time of exercise on option value.

Figure 6.10 shows an in-the-money option value for RAM for varying N while Figure 6.11 shows an out-of-the-money call. I can observe that as the number of time step increases, the option value appears to reache a steady state. From the experiment, analysis of the option values shows that the option values converges (error level set at 0.1%) in 24 steps. Increasing the computation beyond 24 steps did

not yield better solutions for the option values, but only increase the computational cost.

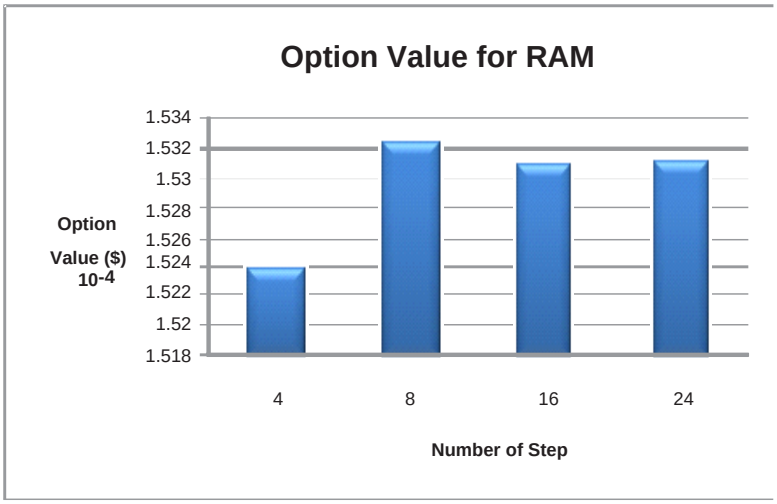

Figure 6.10: Option Value for RAM.

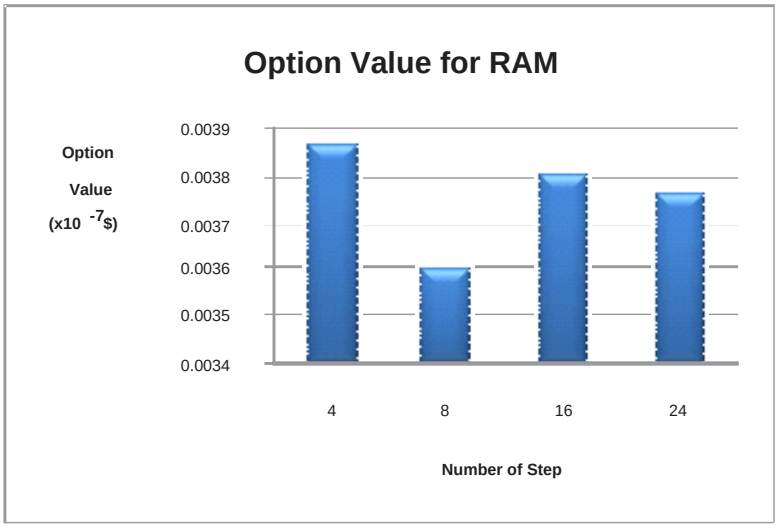

Figure 6.11: Money Option Value for RAM.

In AuverGrid, the number of jobs running increases steadily throughout the year.

In the cases of LCG and SHARCNet, it is likely that some jobs not in the original batch (queue) were completed before their initially estimated completion time. This means that some jobs may unnecessarily wait in the original queue. These original jobs should be compensated by way of (incentive) opportunity. This I achieved using the price variant factor p_f during the option value computation for a specific *gcc*.

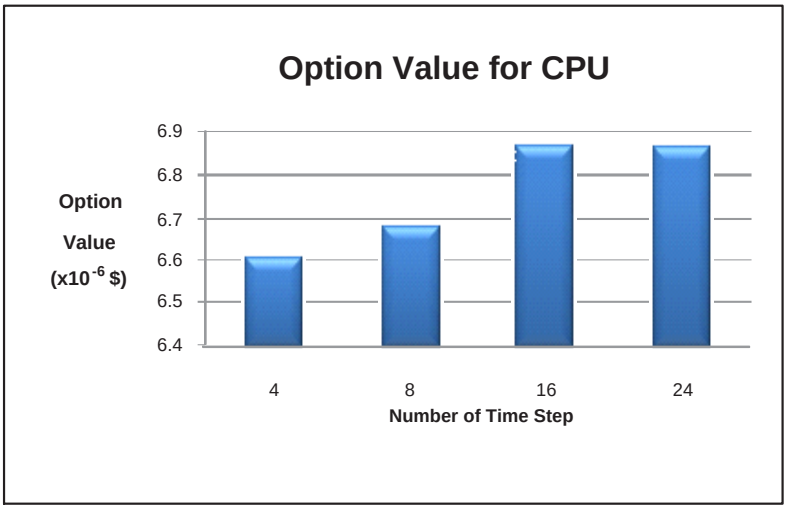

Figure 6.12: Option Value for CPU.

Similarly, I obtain from my simulation the option values for both in-the-money and out-of-the-money CPU using the limits $S = \$68.49$ and $K = \$68.47$ and $\$80.47$ (all values scaled at $(\times 10^{-6})$) and simulated for a varying time step of $4, 8, 16, 24$. Figure 6.12 shows the in-the-money option value for CPU. The option values for in-the-money for other *gccs* under my current study include RAM in Figure 6.10 and HD in Figure 6.13. This behavior shows that at any given time, a users' cost for using the grid resources is the base cost and the extra cost which depends on the time of use of the *gcc* and the value of the p_f depends on changes in the technology or architecture of the grid infrastructure. These variations are unknown before using

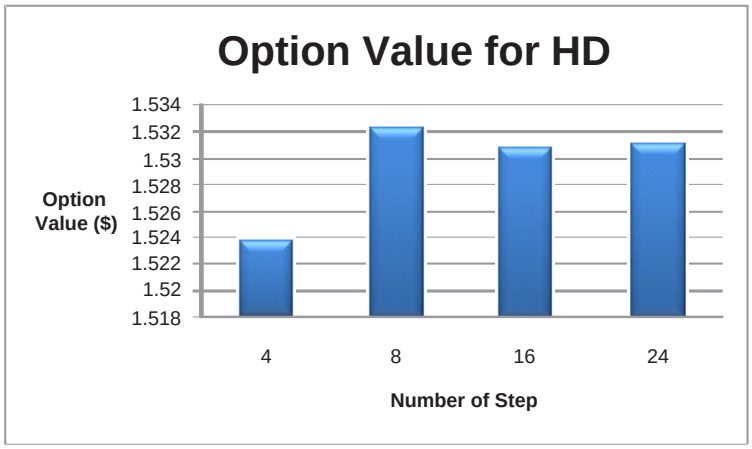

Figure 6.13: Option Value for HD.

the grid resource by exercising the option. Therefore, deciding the exact price of a *gcc* in real life is uncertain and hard to predict. To increase utilization with the same technology, I set the value of p_f to be close to 0.1. With new technology, the value of p_f is set closest to 1.0. Figure 6.3 showed a mapping of the membership function of p_f to time of exercise in the range of fuzzified boundary value of p_f is $[0.1, 1.0]$. My model, therefore, adjusts the price in the use of grid resources by (p_f^{-1}) (for the grid operator) while providing quality service to the user. Figure 6.14 shows a corresponding out-of-the-money option value for CPU. I repeat this experiment for various *gcc*-s. Figure 6.15 shows execution time for HD, CPU, and RAM at various time steps. I stop my computation at step size 24 as reasoned earlier. This is in contrast to the financial market where stock prices are highly volatile and for convergence one needs to see very small step sizes (in other words a large number of steps). The resource provider can benefit for certain values of p_f for which the contract holder will not exercise early. That is, a provider can execute the jobs of

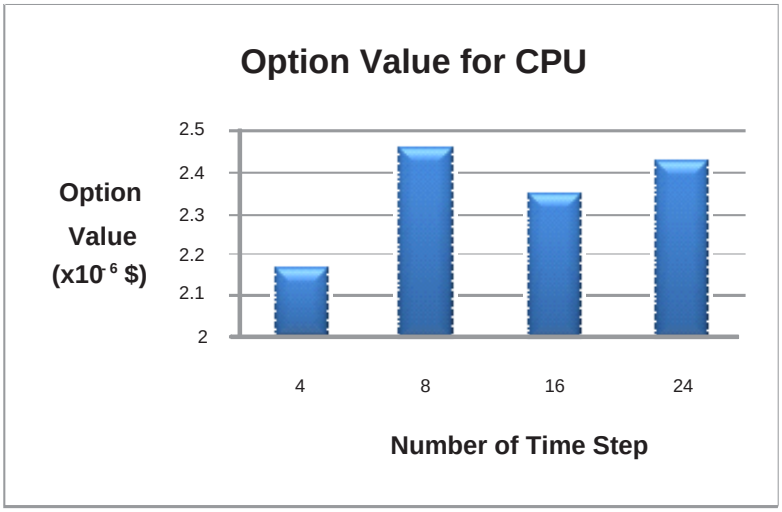

Figure 6.14: Option Value for CPU.

users willing to pay higher prices for the resources. Therefore, the original contract holder still has the time value on his/her option to exercise at a later date. This implies that both the user and the provider can benefit. In other words, the price varying factor p_f helps in achieving the quasi-static equilibrium between the quality of service that the user requires and the profit level (recovery of the expenses, in my thesis) that the service provider would expect. Since the value of p_f is not changed for a given experiment, p_f is not dynamic. Changing the value of p_f dynamically is bit complicated and I leave that issue for future work. Figure 6.16 shows my evaluation for various commodities. In these experiments, I computed the option value for individual *gcc* only. However, I note that most grid jobs use resources in a combination of more than two resources. The correlation among such combined set of *gcc*s is being incorporated into my model to calculate base prices, which will in turn be used to compute the option values for such combinations of *gcc*s.

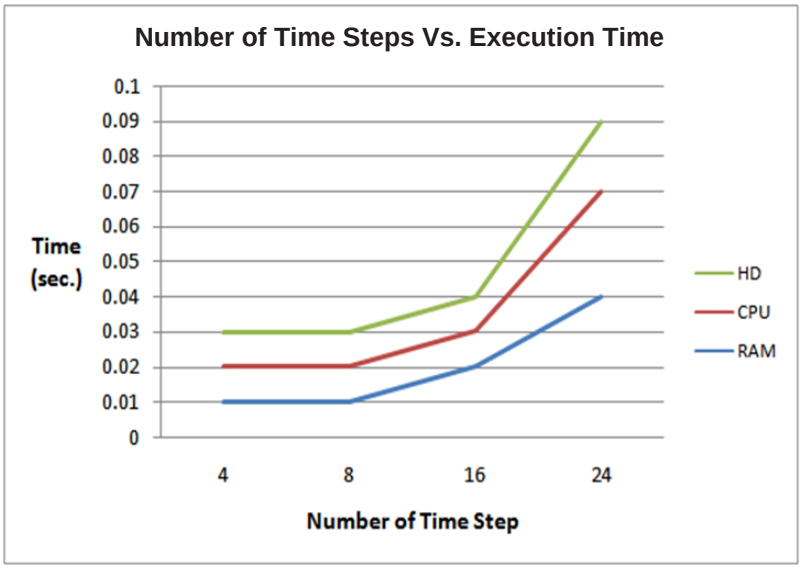

Figure 6.15: Execution Time for Various Commodities.

6.7 Chapter Summary

I provided trace data analysis for two grids (WestGrid and SHARCNet) in Chapter 3. In this chapter, I extended my trace analysis to other four grids (Grid5000, LCG, NorduGrid, and AuverGrid) and emphasized the need for a controller and Commodity Base Prices (CBP). In this chapter, I also presented the design of p_f. Using the CBP and the p_f, I did experiments with trinomial trees in the integrated GridSim simulator to compute option values of various *gccs*.

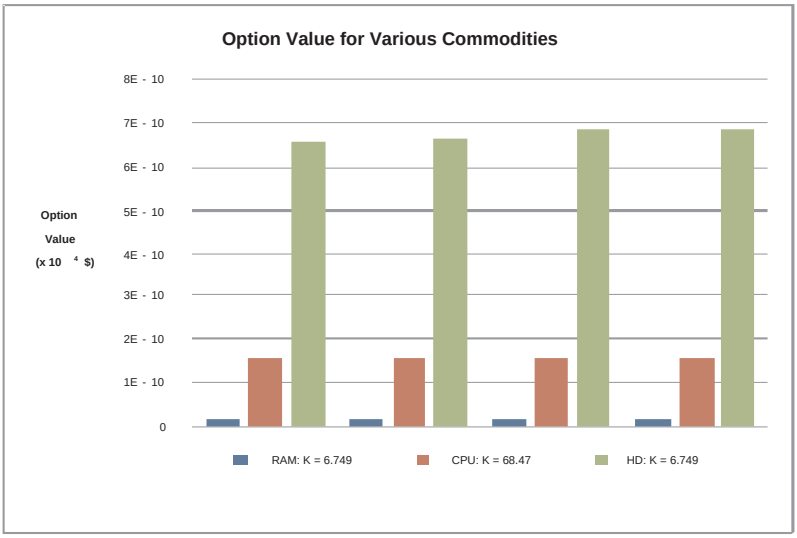

Figure 6.16: Option Value for Various Commodities.

Chapter 7

Conclusions and Future Directions

There have been very few efforts to price grid resources, due to the complex nature of the grid (dispersed ownership, uncertainty in resource availability, resource access policies, absence of standards, and unavailable pricing benchmarks). Moreover, funding provided by government and other agencies make grid resources available for free and has pushed research efforts to price grid resources to secondary importance.

Current interest from businesses and individuals to use grid resources has increased the need for a pricing mechanism. Many grid services that charge a flat rate exist these days. This thesis puts forward an important aspect of grid resource management – resource pricing. I propose a financial options-based pricing model in this thesis. Specifically, three interdisciplinary research threads (Financial Options, real options, and fuzzy logic) form the underlying methodologies for the pricing model.

7.1 Conclusions

In this thesis, I have formulated grid resource pricing as a discrete time financial option pricing problem. Treating grid resources as assets enable me to use financial real options to do the pricing while I use fuzzy logic to provide QoS for the users satisfaction. For the initial problem formulation I used hypothetical data as well as traces from two real grids; WestGrid and SHARCNet to verify the suitability of pricing grid resources using financial options and as an initial proof of concept.

I extended the trace collection and analysis to SHARCNet to validate the observed fluctuations in WestGrid. The results of SHARCNet traces were similar in their fluctuation patterns to WestGrid. My pricing architecture was proposed to consist of a user enhanced SLA document which defines the benefits expressed as QoS (see Appendix C for more details). The third progressive phase of this thesis also proposed to apply a trinomial lattice method for pricing resources and using the same test data as was used in the preliminary studies in [37]. The thesis proposed to balance the usage between under-utilized resources and periods of high usage. The third progressive phase also developed a middleware framework in [44]. The emphasis of the developed framework is the support for increased profitability[1]. The thesis achieved a control of the observed fluctuations by using my designed PVF in the presence of two constraints (from the user (satisfaction) and provider (investment recovery) perspectives), I focused on obtaining a balance which provides a handle to manage flexibility for users and provider. For the provider, my model achieved the following: (a) the provider can determine how many patronage he/she can have

[1]The user's QoS and provider's early recovery on grid infrastructure.

and they by know how long it will take to recover 100% of all the investments in the grid infrastructure. (b) Ability to know a recovery date can also be applied to effective management of the grid resources. That is, the provider is able to know how to control the price of his/her resources in order to stay competitive. The model also provides user flexibility. I use fuzzy logic to capture the user's QoS as a measure of the users' satisfaction while keeping the prices lowing using my price variant function (PVF).

I have also demonstrated that grid resources can be priced by using my Commodity Base Price (CBP). Using the prevailing market prices, I developed CBP which help my model to recover the provider's investment in grid infrastructure in a known time of 2 years. Extending the grid resource utilization trace collection and analysis to other real grids enable me to analyze utilization pattern and provide a level of QoS for the users.

Another contribution of my thesis is in the integration of my model in GridSim toolkit simulator. I incorporated the use of p_f with the trinomial approach in a GridSim simulation to price resources using the discrete time trinomial tree approach and second, the price variant function, p_f, was also incorporated with the reverse Dutch auction in a GridSim simulation. I mapped the time step in the trinomial lattice to the rounds of bidding in the reverse Dutch auction and identify compensations as incentives for the users who are willing to change utilization behavior (willing to exercise later in future).

The thesis objective was to provide a financial option-based model to price grid resources. I use my developed price variant function and the commodity base prices

to maintain an equilibrium between users' expected satisfaction (measured as QoS) for using grid resources and providers expected early recovery on grid infrastructure. I analyzed real grid utilization traces and carried my simulation using GridSim.

7.2 Future Directions

The immediate future work plan is expected to span a period between six and twelve months. Extensions include pricing multiple resources across several grids (Section 7.2.1), incorporating variable commodity base prices into the current model and integrating variable overhead costs (for example building, equipment insurance, staff salaries, equipment upgrades, heating, cooling etc.) into my pricing model (Section 7.2.2). In addition, I plan to immediately extend my current pricing model to cloud services where gccs will be treated as services. That is, a pricing formulated as cloud compute commodities as priced services (IaapS) – a turn from the usual Infrastructure as a service (IaaS) (Section 7.2.3). The life of the long term plan may extend up to 4 years. During this time, I shall extensible capacity of the cloud (resizable compute capacity in the cloud) and green energy resource pricing. I also plan to introduce *"pay-by-barter computing"* – a social pricing scheme that exchange services as a form of payments. Figure 7.1 shows a futuristic plan for the short term and the long term goals of this thesis.

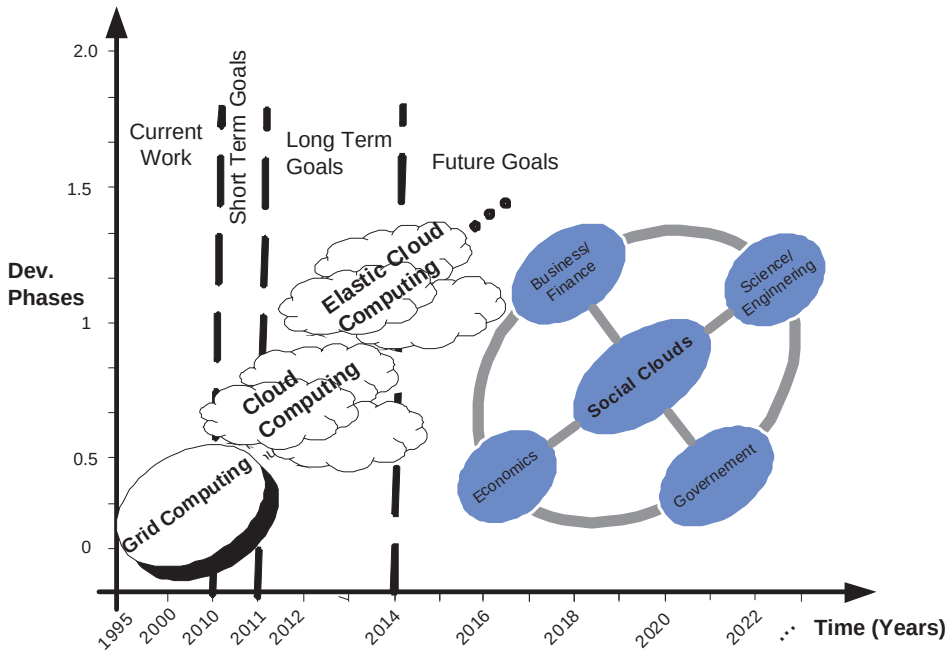

Figure 7.1: Futuristic Overview of the Short term and Long Term Goals.

7.2.1 Multi-Resource/ Service Pricing Across Clouds

The Grid Resources Utilization Matrix (GRUM) provides a mapping of *gcc*-s to various grids. The current research prices a resource of one grid or several resources of the same grid. However, if computations (in real life) require a combination of grid resources across multiple grids or clouds, I shall be faced with the challenge of associating several grids whose characteristic features (ownership, currency, policy, and zonal time) differ. Consider some grids $G_i = g_1, g_2, \cdots, g_n$ and compute commodities that exist across the grids g_{cc} such as $CC_i = cc_1, cc_2, \cdots, cc_m$. Suppose I have set base prices (some assumed base values) such as $P_i = p_1, p_2, \cdots, p_n$, then I can set up a Grid Resources Utilization Matrix (GRUM). For the grid resources use of n grids

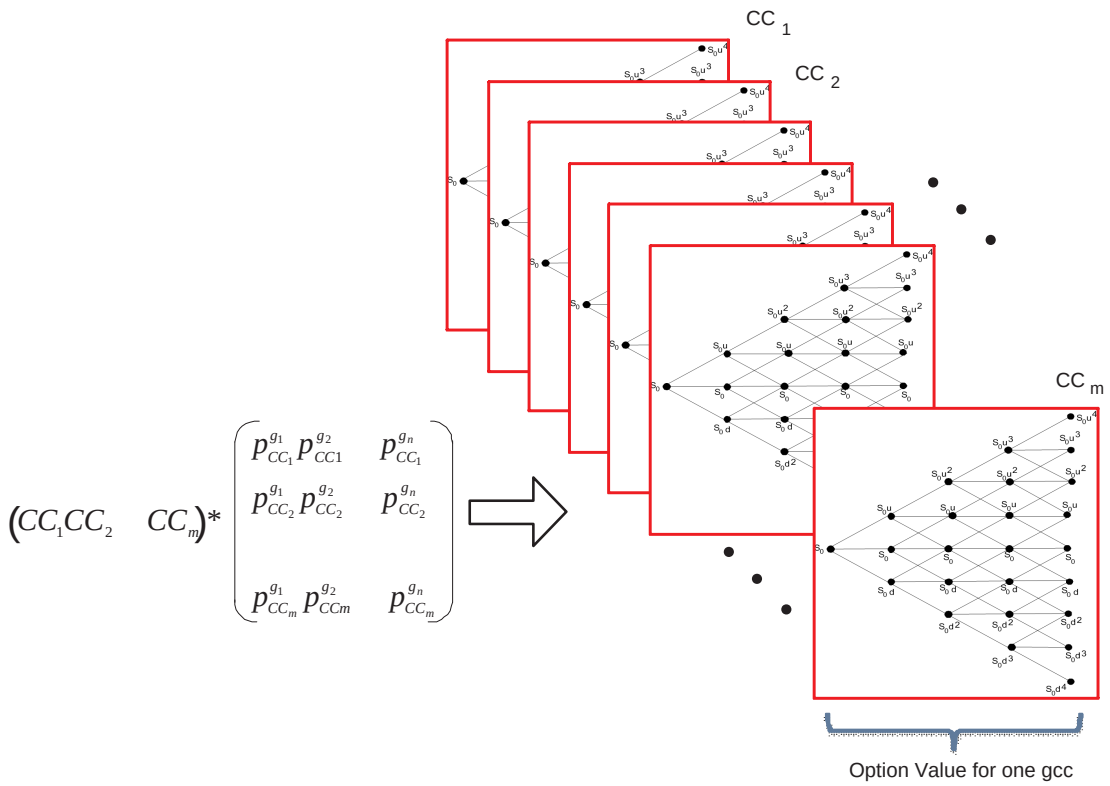

Figure 7.2: Multidimensional Grid and Resource Pricing.

and m commodities and their prices I have:

$$(cc_1 \quad cc_2 \quad \cdots \quad cc_m) \times \begin{pmatrix} p_{cc_1}^{g_1} & p_{cc_1}^{g_2} & \cdots & p_{cc_1}^{g_n} \\ p_{cc_2}^{g_1} & p_{cc_2}^{g_2} & \cdots & p_{cc_2}^{g_n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{cc_m}^{g_1} & p_{cc_m}^{g_2} & \cdots & p_{cc_m}^{g_n} \end{pmatrix} \tag{7.1}$$

I treat each instance of $[{cc_m}^{g_n} * p_{cc_m}^{g_n}]$ in Equation (7.1) as a trinomial tree whose solution requires large computational resources of the grid because of its large size. Figure 7.2 idealizes the pricing multiple resources across multiple grids. A further

challenge will include the provision of QoS. In multiple grids such as the GRUM, QoS is provided at the global level rather than at the individual local QoS as in the thesis work.

7.2.2 Integrating I-O Analysis and Variable Commodity Base Prices

As additional future work, I will consider the set of inputs to a computational grid which are necessary to execute user jobs to price grid resources. These sets of inputs consist of (i) fixed costs of acquiring computational resources (CPU cycles, memory, network bandwidths, computing power, disks, processor times, and various visualization tools, software and specialized instruments) that exist as non-storable compute cycles (grid compute commodities) and (ii) variable overhead costs (building, fixed financing, equipment insurance, salaries, equipment upgrades, heating, cooling). The interaction of fixed costs and variable costs must be balanced for the grid to achieve its objective. Since obtaining a balance (especially in multiple grids) is hard, I will apply Leontiff's Input-Output Economics [45] (Input-Output Analysis (IOA)) concept to capture the dependencies that exist between the computational costs (i.e., the price of components for executing jobs in a grid) and the various inputs (fixed and variable costs) to the grid. Since I know that fixed and variable cost of grid infrastructure across multiple grids can determine the available grid resources, I will set base prices using the activities within a specific grid instead of market trends. I will consider applying Activity-Based Costing (ABC) [9] to determine the base costs of resources.

7.2.3 Pricing and Management of Cloud Compute Resources

The Cloud computing paradigm involves delivering of hosted services over the Internet. These services are broadly divided into three categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). A cloud service is elastic (resizable compute capacity in the cloud [122]) and fully managed by the provider. It can be sold on demand. A cloud can be private or public. A public cloud sells services to anyone on the Internet. A private cloud is a proprietary network or a data center that supplies hosted services to a limited number of people. When a service provider uses public cloud resources to create its private cloud, the result is called a virtual private cloud. In the future, I plan to move toward modeling enterprize service architecture for provision of cost, benefits, and planned management for cloud based services. The model will consist of the cloud services, data, infrastructure, management, processes, quality assurance, and flexible pricing achieved using real options. These various components will be modeled as a priced Service, that is, Infrastructure-as-a-priced-Service (IaapS), Platform-as-a-priced-Service (PaapS) and Software-as-a-priced-Service (SaapS).

7.2.4 Pricing Green Energy

Another extension of my current research will focus on the pricing and management of green energy resources. Currently, research efforts in energy optimization such as green energy have only considered waste management generally in terms of cutting down spending (billing) [123]. While I focus on resource and energy management for the cloud, I will specifically examine pricing green energy, which, if neglected,

may soon form a significant source of spending in data centers. Some major cloud providers such as Google are estimated to be spending over $35 Million dollars on electricity bills each year [124]. From the perspective of resource pricing, I believe that there is a lot to be discovered. I intend to spend some time to explore the benefits and applications of financial options to pricing green energy as well as managing the resources.

Bibliography

[1] Canarie, "Canada's Advanced Research and Innovation Network," June 2010. [Online]. Available: http://www.canarie.ca/.

[2] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Int'l. Journal High Performance Computing Applications*, vol. 15, no. 3, pp. 200–222, 2001.

[3] R. Buyya and M. M. Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing," *Journal of Concurrency and Computation: Practice and Experience (CCPE)*, vol. 14, no. 13, pp. 1175–1220, 2002.

[4] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco, USA: Morgan Kaufmann Publishers, Inc., 1999.

[5] P. M. Lyster, L. Bergman, P. Li, D. Stanfill, B. Crippe, R. Blom, and D. Okaya, "CASA Gigabit Supercomputing Network: CALCRUST Three-Dimensional Real-Time Multi-Dataset Rendering," in *Proc. of Supercomputing*, Minneapolis, November 1992.

[6] R. Buyya and S. Venugopal, "A Gentle Introduction to Grid Computing and Technologies," *Computer Society of India (CSI)*, vol. 1, no. 29, pp. 9–19, 2005.

[7] S. Pantry and P. Griffiths, *The Complete Guide to Preparing and Implementary Service Level Agreements*. Library Association Publishing, 1st. Edition, 1997.

[8] R. Krishnan, "Grid Economics: A Selective Discussion of Two Research Problems," *Grid Computing*, vol. 6, no. 3, pp. 219–224, 2008.

[9] R. S. Kaplan and R. Cooper, *Cost and Effect: Using Integrated Cost Systems to Drive Profitability and Performance*. Harvard Business School Press, 1998.

[10] AppNexus. (2010, June) Appnexus. [Online]. Available: http://www.appneus.com/

[11] GoGrid. (2010, June) Cloud Hosting: Instant Windows and Linux Cloud Servers. [Online]. Available: http://www.gogrid.com/

[12] Joyent. (2010, June) Accelerator. [Online]. Available: http://joyent.com/accelerator/

[13] Rack Space. (2010, July) Rack Space Hosting. [Online]. Available: http://www.rackspace.com/index.php

[14] Google. (2010, June) App engine. [Online]. Available: http://code.google.com/intl/de-DE/appengine/

[15] Amazon. (2010, December) Amazon Web Services. [Online]. Available: http://aws.amazon.com/

[16] Microsoft Corporation. (2010, June) Azure services platform. [Online]. Available: http://code.google.com/intl/de-DE/appengine/

[17] M. R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel, "Amazon S3 for Science Grids: A Viable Solution?" in *DADC '08: Proc. of the 2008 Int'l. workshop on Data-aware distributed computing.* New York, NY, USA: ACM, 2008, pp. 55–64.

[18] Amazon.com, "Simple storage system," June 2010. [Online]. Available: http://aws.amazon.com/s3/.

[19] A. Caracas and J. Altmann, "A Pricing Information Service for Grid Computing," in *Proc. of the 5th Int'l. Workshop on Middleware for Grid Computing (MGC '07).* New York, NY, USA: ACM, 2007, pp. 1–6.

[20] X. Zhang and Y. Dong, "Optimizing Xen VMM Based on Intel®Virtualization Technology," in *Proc. of the 2008 Int'l. Conf. on Internet Computing in Science and Engineering (ICICSE '08).* Leipzig, Germany: IEEE Computer Society, 2008, pp. 367–374.

[21] J. C. Hull, *Options, Futures, and Other Derivatives.* Prentice Hall, 7th Edition, 2009.

[22] A. K. Dixit and R. S. Pindyck, "The Options Approach to Capital Investment," in *The Economic Impact of Knowledge*, D. Neef, G. A. Siesfeld, and J. Cefola, Eds. Boston: Butterworth-Heinemann, 1998, pp. 325–340.

[23] A. B. Abel, A. K. Dixit, J. C. Eberly, and R. S. Pindyck, "Options, the Value

of Capital, and Investment," *The Quarterly Journal of Economics, MIT Press*, vol. 111, no. 3, 1996.

[24] T. P. Morgan, "World IT Spending Estimate," *Windows News and Insight*, vol. 8, no. 5, 2008.

[25] D. M. Chance and P. P. Peterson, *Real Options and Investment Valuation*. The Research Foundation of AMIR, 1st. Edition, 2002.

[26] L. Zadeh, "A Fuzzy Set-Theoretic Interpretation of Linguistic Hedges," *Journal of Cybernetics*, vol. 3, no. 2, pp. 4–34, 1972.

[27] D. Dubois and H. Prade, "What are Fuzzy Rules and How to Use Them," *Fuzzy Sets Systems*, vol. 84, no. 2, pp. 169–189, 1996.

[28] S. Mitaim and B. Kosko, "What is the Best Shape for a Fuzzy Set in Function Approximation?" in *Proc. of the 5th IEEE Int'l. Conf. on Fuzzy Systems*, vol. 2. New Orleans, LA USA: Kluwer, B.V., 1996, pp. 1237–1243.

[29] T. J. Ross, *Fuzzy Logic with Engineering Applications*. John Wiley and Sons Inc, 1st Edition, 2004.

[30] G. Bojadziev and M. Bojadziev, *Fuzzy Logic for Business, Finance, and Management Modeling*. World Scientific Press, 2nd Edition, 2007.

[31] A. Iosup, D. Catalin, E. Dick, L. Hui, and W. Lex, "How Are Real Grids Used? The Analysis of Four Grid Traces and its Implications," in *Proc. of the 7th IEEE/ACM Int'l. Conf. on Grid Computing (GRID '06)*. Barcelona: IEEE Computer Society, 2006, pp. 262–269.

[32] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, "A security architecture for computational grids," in *ACM Conf. on Computer and Communications Security*, 1998, pp. 83–92.

[33] R. Buyya, M. Murshed, and D. Abramson, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing," *Journal of Concurrency and Computation: Practice and Experience (CCPE) Issue 13-15*, vol. 14, no. 13, pp. 1175–1220, 2002.

[34] W. Smith, I. Foster, and V. Taylor, "Scheduling with advanced reservations," in *Proc. of the Int'l. parallel and Distributed Processing Symposium*, Cancum Mexico, May 2000.

[35] A. Sulistio, W. Schiffmann, and R. Buyya, "Advanced Reservation-Based Scheduling of Task Graphs on Clusters," in *Proc. of the 13th IntI Conf. on High Performance Computing (HiPC), Bangalore, India, Dec. 18-21*, 2006.

[36] K. M. Sim, "Grid Commerce, Market-Driven G-Negotiation, and Grid Resource Management," *IEEE Transactions Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 36, no. 6, 2004.

[37] D. Allenotor and R. K. Thulasiram, "A Grid Resources Valuation Model Using Fuzzy Real Option," in *Proc. 5th Int'l. Symposium on Parallel and Distributed Processing and Applications. (ISPA-07)*, I. Stojmenovic, R. K. Thulasiram, L. T. Yang, W. Jia, M. Guo, and R. F. de Mello, Eds. Niagara Falls, ON, CANADA: Springer, 2007.

[38] grid3.com, "Grid3 International," June 2010. [Online]. Available: http://www.grid3.com/.

[39] T. G. Johan, J. Montagnat, and I. Cnrs, "An Experimental Comparison of Grid5000 Clusters and the EGEE grid," in *In Workshop on Grid*, 2006.

[40] LCG-LHC, "Large Hadron Collider (LHC)," June 2010. [Online]. Available: http://lcg.web.cern.ch/LCG/New/index.html

[41] SHARCNet, "Shared hierarchical academic research computing network (sharcnet)." June 2010. [Online]. Available: http://www.sharcnet.ca/Performance/cur_perf.php.

[42] WestGrid, "Western Canada Reserach Grid," June 2010. [Online]. Available: http://www.westgrid.ca/home.html.

[43] NorduGrid, "Nordugrid," June 2010. [Online]. Available: http://www.nordugrid.org/.

[44] D. Allenotor and R. K. Thulasiram, "G-FRoM: Grid Resources Pricing A Fuzzy Real Option Model," in *Proc. 3rd Int' Conf. on e-Science and Grid Computing (eScience 2007) Bangalore, India, December 10-13*, 2007.

[45] W. Leontief, *Input-Output Economics*, 2nd ed. Oxford University Press, 1986.

[46] D. Allenotor and R. K. Thulasiram, "Grid Resources Pricing: A Novel Financial Option Based Quality of Service-profit Quasi-static Equilibrium Model," in *Proc. of the 2008 9th IEEE/ACM Int'l. Conf. on Grid Computing (GRID '08)*. Tsukuba, Japan: IEEE Computer Society, 2008, pp. 75–84.

[47] D. Allenotor, R. Thulasiram, and P. Thulasiraman, "A Financial Option Based Grid Resources Pricing Model: Towards an Equilibrium Between Quality of Service for Users and Profitability for Service Providers," in *4th IEEE Intl. Conf. on Grid and Pervasive Computing (GPC 2009)*, N. Abdennadher and D. Petcu, Eds. Geneva, Switzerland: Springer, 2009, pp. 13–24.

[48] Grid5000, "Grid5000," June 2010. [Online]. Available: https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home.

[49] AuverGrid, "Auvergrid," June 2010. [Online]. Available: http://www.auvergrid.fr/.

[50] DAS-2, "The Distributed ASCI Supercomputer 2 (DAS-2)," June 2010. [Online]. Available: http://www.cs.vu.nl/das2/.

[51] R. Buyya, D. Abramson, and J. Giddy, "A Case for Economy Grid Architecture for Service Oriented Grid Computing," in *Proc. of the 10th Heterogeneous Computing Workshop (in IPDPS '01)*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 4–18.

[52] R. Buyya, D. Abramson, and S. Venugopal, "The grid economy," in *IEEE Journal*, 2005, pp. 698 – 714.

[53] C. S. Yeo and R. Buyya, "Integrated risk analysis for a commercial computing service," in *Proc. of the 21st IEEE Int'l. Parallel and Distributed Processing Symposium (IPDPS 2007, IEEE CS Press, LosAlamitos, CA, USA)*, 2007.

[54] R. Wolski, J. Plank, J. Brevik, and T. Bryan, "Analyzing market-based resource allocation strategies for the computational grid," *Int. Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 285–281, 2001.

[55] H. K. Bhargava and S. Sundaresan, "Computing as Utility: Managing Availability, Commitment, and Pricing Through Contingent Bid Auctions," *Journal of Management Information Systems*, pp. 201–227, 2004.

[56] W. Kang, H. H. Huang, and A. Grimshaw, "A Highly Available Job Execution Service in Computational Service Market," in *Proc. of the 8th IEEE/ACM Int'l. Conf. on Grid Computing (GRID '07).* Austin Texas, USA: IEEE Computer Society, 2007, pp. 275–282.

[57] Z. Tan and J. R. Gurd, "Market-based Grid Resource Allocation Using a Stable Continuous Double Auction," in *Proc. of the 8th IEEE/ACM Int'l. Conf. on Grid Computing. (Grid '07).* Austin Texas, USA: IEEE Computer Society, September 19-21 2007, pp. 283 – 290.

[58] I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt, and A. Roy, "A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation," in *Proc. of the 7th Int'l. Workshop on Quality of Service (IWQoS '99)*, 1999, pp. 27–36.

[59] L. Chunlin and L. Layuan, "Pricing and Resource Allocation in Computational Grid with Utility Functions," in *Int'l. Conf. on Information Technology: Coding and Computing (ITCC'05).*, vol. II, September 2005.

[60] A. Sulistio, W. Schiffmann, and R. Buyya, "Using Revenue Management to Determine Pricing of Revervations," in *Proc. 3rd Int'l. Conf. on e-Science and Grid Computing (eScience 2007)*. Bangalore, India: IEEE Computer Society, December 10-13 2007, pp. 396–405.

[61] V. Krishna and M. Perry, "Efficient Mechanism Design," Washington University in St. Louis, EconWPA, The Economics Working Paper Archive, Game Theory and Information 9703010, March 1998.

[62] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *The Int'l. Journal of Supercomputer Applications and High Performance Computing*, vol. 11, no. 2, 1997.

[63] ——, "The Globus Project: A Status Report," *Future Generation Computer Systems*, vol. 15, no. 5-6, pp. 607–621, 1999.

[64] GLOBUS, "http://www.globus.org," 2007.

[65] B. Jacob, L. Ferreira, N. Bieberstein, C. Gilzean, J. Girard, R. Strachowski, and S. Yu, *Enabling Applications for Grid Computing with Globus*. IBM Corp., 1st Edition, 2003.

[66] Globus, "Globus toolkit," December 2010. [Online]. Available: http://www.globus.org/toolkit/

[67] E. Elmroth, P. Gardfjll, O. Mulmo, and T. Sandholm, "An OGSA-based Bank Service for Grid Accounting Systems," in *in Applied Parallel Computing, Lecture Notes in Computer Science*. Springer Verlag, 2006, pp. 1051–1060.

[68] R. Buyya and S. Venugopal, "The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report," in *1st IEEE Int'l. Workshop Grid Economics and Business Models GECON'04*. Seoul, Korea: IEEE Press, New Jersey, USA, 2004, pp. 19–66.

[69] A. Barmouta and R. Buyya, "GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration," in *IPDPS '03: Proc. of the 17th Int'l. Symposium on Parallel and Distributed Processing*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 24–51.

[70] M. J. Litzkow, M. Livny, and M. W. Mutka, "Condor: A Hunter of Idle Workstations," in *ICDCS '88: Proc. of 8th Int'l. Conf. of Distributed Computing Systems*. Los Alamitos, California: IEEE Computer Society Press, 1988.

[71] D. H. J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne, "A Worldwide Flock of Condors: Load Sharing Among Workstation Clusters," *Future Generation Computer Systems*, vol. 12, no. 1, pp. 53–65, 1996.

[72] A. S. Grimshaw, W. A. Wulf, J. C. French, A. C. Weaver, and P. F. R. Jr., "Legion: The Next Logical Step Toward a Nationwide Virtual Computer," Computer Science Department, University of Virginia, Charlottesville, VA, USA, Tech. Rep., 1994.

[73] K. Appleby, S. Fakhoury, L. Fong, G. Goldszmidth, M. Kalantar, S. Krishnakumar, D. P. Pazel, J. Pershing, and B. Rochwerger, "Océano – SLA Based Management of a Computing Utility," in *Proc. of the 7th IFIP/IEEE Sympo-*

sium on Integrated Network Management (IM2001). Washington, DC, USA: IEEE Computer Society, 2001, pp. 855–868.

[74] D. Abramson, J. Giddy, and L. Kotler, "High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?" in *IPDPS '00: Proc. of the 14th Int'l. Symposium on Parallel and Distributed Processing.* Washington, DC, USA: IEEE Computer Society, 2000, pp. 520–528.

[75] R. Buyya, J. Giddy, and D. Abramson, "An Evaluation of Economy-based Resource Trading and Scheduling on Computational Power Grids for Parameter Sweep Applications," in *Proc. of the 2nd Workshop on Active Middleware Services (AMS 2000), (In conjunction with HPDC 2001).* Pittsburgh, PA, USA: Kluwer Academic Press, 2000.

[76] D. Abramson, P. Roe, L. Kotler, and D. Mather, "ActiveSheets: Supercomputing with Spreadsheets." in *HPC '01: Proc. of the High Performance Computing Symposium, Advanced Simulation Technologies Conf.* San Diego, California, USA: Society for Modeling and Simulation (SCS) Press, 2001, pp. 110–115.

[77] G. Fabrizio and M.-E. Begin, "EGEE - Providing a Production Quality Grid for e-Science," in *Proc. of the 2005 IEEE Int'l. Symposium on Mass Storage Systems and Technology, Local to Global Data Interoperability (LGDI '05) - Challenges and Technologies.* IEEE Computer Society, 2005, pp. 88–92.

[78] J. L. Vázquez-Poletti, E. Huedo, R. S. Montero, and I. M. Llorente, "Coor-

dinated Harnessing of the IRISgrid and EGEE Testbeds with Gridway," *J. Parallel Distributed Computing*, vol. 66, no. 5, pp. 763–771, 2006.

[79] GRIA, "DataGrid Accounting System (DAGAS)," July 2009. [Online]. Available: https://edms.cern.ch/file/571271/1/EGEE-DGAS-PA-Guide.pdf.

[80] Nimrod/G, "http://www.csse.monash.edu.au/ davida/nimrod/nimrodg.htm," 2010.

[81] NASA-PG, "http://www.nas.nasa.gov/about/ipg/ipg.html," 2010.

[82] CERN, "http://public.web.cern.ch/public/welcome.html," 2010.

[83] WestGrid, "Western Canada Reserach Grid," June 2010. [Online]. Available: http://www.westgrid.ca/home.html.

[84] GRIA, "Grid Resources for Industrial Applications," June 2010. [Online]. Available: http://www.gria.org.

[85] S. Mike, T. Steve, D. R. David, and Z. Ed, "Experiences with GRIA — Industrial Applications on a Web Services Grid," in *Proc. of the 1st Int'l. Conf. on e-Science and Grid Computing (e-Science '05)*. Melbourne, Australia: IEEE Computer Society, Dec. 5 - 8 2005, pp. 98–105.

[86] Amazon, "Amazon Elastic Compute Cloud (Amazon EC2)," January 2010. [Online]. Available: http://aws.amazon.com/ec2/.

[87] A. Takefusa, S. Matsuoka, K. Aida, H. Nakada, and U. Nagashima, "Overview of a Performance Evaluation System for Global Computing Scheduling Algo-

rithms," in *Proc. of the 8th IEEE Int'l. Symposium on High Performance Distributed Computing (HPDC 1999)*. Redondo Beach, California, USA: IEEE Computer Society, July 1999.

[88] A. Legrand, L. Marchal, and H. Casanova, "Scheduling Distributed Applications: The SimGrid Simulation Framework," in *Proc. of the 3rd Int'l. Symposium on Cluster Computing and the Grid (CCGRID '03)*. Tokyo, Japan: IEEE Computer Society, 2003, pp. 138–145.

[89] A. Sulistio, U. Cibej, S. Venugopal, B. Robic, and R. Buyya, "A Toolkit for Modelling and Simulating Data Grids: An Extension to GridSim," *Journal of Concurrency and Computation: Practice and Experience (CCPE)*, vol. 20, no. 13, pp. 1591–1609, 2008.

[90] C. L. Dumitrescu and I. Foster, "GangSim: A Simulator for Grid Scheduling Studies," in *Proc. of the 5th IEEE/ACM Int'l. Symposium on Cluster Computing and the Grid (CCGrid'05) - Volume 2*. Cardiff, UK: IEEE Computer Society, 2005, pp. 1151–1158.

[91] W. H. Bell, D. G. Camerona, L. Capozza, P. A. Millar, K. Stockinger, and F. Zini, "OptorSim - A Grid Simulator for Studying Dynamic Data Replication Strategies," *Int'l. Journal of High Performance Computing Applications*, vol. 17, no. 4, pp. 403–416, 2003.

[92] M. Feldman, K. Lai, and L. Zhang, "A Price-anticipating Resource Allocation Mechanism for Distributed Shared Clusters," in *Proc. of the 6th ACM Conf. on*

Electronic Commerce (EC 2005),. Vancouver, BC, Canada: ACM, 2005, pp. 127–136.

[93] C. Daskalakis, G. Schoenebeck, G. Valiant, and P. Valiant, "On the Complexity of Nash Equilibria of Action-Graph Games," in *Proc. of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '09)*. New York Marriott Downtown, New York, New York: Society for Industrial and Applied Mathematics, 2009, pp. 710–719.

[94] M. P. Wellman, W. E. Walsha, P. R. Wurman, and J. K. MacKie-Mason, "Auction Protocols for Decentralized Scheduling," *Jornal of Games and Economic Behavior*, vol. 35, no. 1–2, pp. 271–303, 2001.

[95] K. Lai, L. Rasmusson, E. Adar, L. Zhang, and B. A. Huberman, "Tycoon: An Implementation of a Distributed, Market-based Resource Allocation System," *Multiagent Grid Syst.*, vol. 1, no. 3, pp. 169–182, 2005.

[96] D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: The Condor Experience," *Journal of Concurrency and Computation: Practice and Experience*, vol. 17, no. 2-4, pp. 323–356, 2005.

[97] A. W. Mu'alem and D. G. Feitelson, "Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling," *IEEE Transactions on Parallel Distributed Systems*, vol. 12, no. 6, pp. 529–543, 2001.

[98] W. Tang, N. Desai, D. Buettner, and Z. Lan, "Analyzing and Adjusting User

Runtime Estimates to Improve Job Scheduling on Blue Gene/P,," in *Proc. of IEEE Int'l. Parallel and Distributed Processing Symposium (IPDPS'10)*, 2010.

[99] A. Mutz, R. Wolski, and J. Brevik, "Eliciting Honest Value Information in a Batch-Queue Environment," in *The 8th IEEE/ACM IntI Conf. on Grid Computing.* Austin, Texas, USA.: IEEE Press, September 2007.

[100] W. Kang, H. H. Huang, and A. Grimshaw, "A Highly Available Job Execution Service in Computational Service Market," in *The 8th IEEE/ACM IntI Conf. on Grid Computing. (Grid 2007).* Austin Texas, USA: IEEE Computer Society, September 19-21 2007.

[101] F. Black and Scholes, "The Pricing of Options and Corporate Liabilities," *Journal of Political Economy*, vol. 81, no. 3, pp. 637–659, 1973.

[102] R. C. Merton, "Theory of Rational Option Pricing," *Bell Journal of Economics*, vol. 4, no. 1, pp. 141–183, 1973.

[103] P. P. Boyle, "Options: A Monte Carlo Approach," *Journal of Financial Economics*, vol. 4, no. 3, pp. 323–338, May 1977.

[104] S. Rahmail, I. Shiller, and R. K. Thulasiram, "Different Estimators of the Underlying Asset's Volatility and Option Pricing Errors: Parallel Monte-Carlo Simulation Integrated Risk Analysis for a Commercial Computing Service," in *Proc. of the Int'l. Conf. on Computational Finance and its Applications (ICCFA).* Bologna, Italy: ACM Computer Society, 2004, pp. 121–131.

[105] J. C. Cox, S. A. Ross, and M. Rubinstein, "Option pricing: A Simplified Approach," *Journal of Financial Economics*, vol. 3, no. 7, pp. 229–263, 1979.

[106] R. K. Thulasiram, L. Litov, H. Nojumi, C. T. Downing, and G. R. Gao., "Multithreaded Algorithms for Pricing a Class of Complex Options," in *Proc. (CD-ROM) of the 15th IEEE/ACM Int'l. Parallel and Distributed Processing Symposium (IPDPS), Anchorage (Alaska) USA, April*, 2001.

[107] S. Barua, R. K. Thulasiram, and P. Thulasiraman, "High Performance Computing for a Financial Application Using Fast Fourier Transform," in *Proc. European Parallel Computing Conf., (EuroPar2005), Lisbon, Portugal, Aug.30-Sep2*, August 2005, pp. 1246–1253.

[108] D. Tavalla and C. Randall, *Pricing Financial Instruments: The Finite Difference Method.* John Wiley and Sons, New York, NY, 2000.

[109] R. K. Thulasiram and P. Thulasiraman, "Performance Evaluation of a Multithreaded Fast Fourier Transform Algorithm for Derivative Pricing," *The Journal of Supercomputing*, vol. 26, no. 1, pp. 43–58, 2003.

[110] R. K. Thulasiram, C. Zhen, A. Chhabra, P. Thulasiraman, and A. Gumel, "A Second Order L_0 Stable Algorithm for Evaluating European Options," *Int. Journal of High Performance Computing and Networking (IJHPCN)*, no. 5-6, pp. 311–320, 2006.

[111] M. Amico, Z. J. Pasek, F. Asl, and G. Perrone., "Simulation Methodology for Collateralized Debt and Real Options: A New Methodology to Evaluate the

Real Options of Investment Using Binomial Trees and Monte Carlo Simulation," in *Proc. of the 35th Conf. on Winter Simulation (WSC '03)*. New Orleans, Louisiana: Winter Simulation Conference, 2003.

[112] A. Goolsbee and P. J. Klenow, "Valuing consumer products by the time spent using them: An application to the internet," National Bureau of Economic Research, Inc, NBER Working Papers, Tech. Rep. 1995, August 2006.

[113] A. A. Gray, P. Arabshahi, E. Lamassoure, C. Okino, and J. Andringa, "A real option framework for space mission design," NASA, LAB, Tech. Rep. Jet Propulsion labouratory, August 2004.

[114] C. Carlsson and R. Fullér, "A fuzzy approach to real option valuation," *Journal of Fuzzy Sets and Systems*, vol. 139, no. 2, pp. 297–312, 2003.

[115] A. Gupta, L. Zhang, , and S. Kalyanaraman, "Simulation for Risk Management: A Two-component Spot Pricing Framework for Loss-rate Guaranteed Internet Service Contracts," in *Proc. of the 35th Conf. on Winter simulation (WSC'03)*. Winter Simulation Conference, 2003, pp. 372–380.

[116] Y. d'Halluin, P. A. Forsyth, and K. R. Vetzal, "Managing capacity for telecommunications networks under uncertainty," *Journal of IEEE/ACM Transaction on Networking*, vol. 10, no. 4, 2002.

[117] Parallel-Workloads-Archive, "Logs of Real Parallel Workloads," March 2010. [Online]. Available: http://www.cs.huji.ac.il/labs/parallel/workload/logs.html.

[118] Canarie, "Investment and Infrastructure in Support of Innovation:

Annual Report to the Minister," November 2009. [Online]. Available: http://www.canarie.ca/templates/about/publications/docs/areportminister_2009.pdf.

[119] D. J. Higham, *An Introduction to Financial Option Valuation: Mathematics, statistics and computation.* Cambridge, First Edition, 2004.

[120] R. Bolze, F. Cappello, E. Caron, M. Daydé, F. Desprez, E. Jeannot, Y. Jégou, S. Lanteri, J. Leduc, N. Melab, G. Mornet, R. Namyst, P. Primet, B. Quetier, O. Richard, E.-G. Talbi, and T. Irena, "Grid'5000: A Large Scale and Highly Reconfigurable Experimental Grid Testbed," *Int'l. Journal of High Performance Computing Applications (IJHPCA)*, vol. 20, no. 4, pp. 481–494, 2006.

[121] NorduGrid, "Advanced Resource Connector (ARC) Grid Monitor," June 2010. [Online]. Available: http://www.nordugrid.org/monitor/loadmon.php.

[122] D. Cui, Y. Wu, and Q. Zhang, "Massive Spatial Data Processing Model Based on Cloud Computing Model," in *Proceedings of the 2010 Third Int'l. Joint Conf. on Computational Science and Optimization - Volume 02*, ser. CSO '10. Washington, DC, USA: IEEE Computer Society, 2010.

[123] A. T. Velte, T. J. Velte, and R. Elsenpeter, *Cloud Computing: A Practical Approach.* McGraw Hill, First Edition, 2010.

[124] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the Electric Bill for Internet-scale Systems," in *Proc. of ACM Special Interest Group on Data Communications (SIGCOMM '09).* Spain, Barcelona: ACM Computer Society, 2009, pp. 123–134.

List of Abbreviations

ABC	Activity Based Costing
APIs	Application Program Interfaces
CASA	Collaborative Adaptive Sensing of the Atmosphere
CBP	Commodity Base Price
CDA	Continuous Double Auction
DCF	Discounted Cash Flow
DGAS	Distributed Grid Accounting System
EC2	Elastic Compute Cloud
EGEE	Enabling Grid for EsciencE
GARA	Globus Advanced Reservation and Allocation
gcc	Grid Compute Commodity
G-FRoM	Grid Resources Pricing - A Fuzzy Real option Model
GGF	Global Grid Forum

GRIA	Grid Resources for Industrial Applications
GSI	Grid Security Infrastructure
GSP	Grid Service Provider
GT	Globus Toolkit
GT4	Globus Toolkit version 4:0
HPC	High Performance Computing
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IOA	Input-Output Analysis
LDAP	Lightweight Directory Access Protocol
LHC	Large Hadron Collider
NPV	Net Preset Value
OASIS	Organization for the Advancement of Structured Information Standards
OGSI	Open Grid Services Infrastructure
PVF	Price Variant Function
QoS	Quality of Service
RaaS	Resource as a Service

S3	Simple Service Storage
SCDA	Stable Continuous Double Auction
SGAS	Swedish Grid Accounting System
SHARCNet	Shared Hierarchical Academic Research Computing Network
SLA	Service Level Agreement
SOAP	Simple Object Access Protocol
SQS	Simple Queue Service
SSL	Secure Socket Layer
TLS	Transport Layer Security
W3C	World Wide Web Consortium

Appendix A

Input Output Analysis (IOA) and Activity Based Costing (ABC)

Input Output (I-O) analysis is a process whereby inputs from one sector of an industry or economy produce outputs for consumption or for inputs for another sector of industry or economy. The I-O table captures the transactions between various participating economies. With the I-O table, one can estimate the change in demand for inputs resulting from a change in production of the final good. IOA [45] is the analysis of the interactions and the interdependencies between a systems input and output operations. It identifies the perfect mix of project resources (raw material, labor, and capital) – called the inputs that is required to achieve the desired production goals – called the outputs. The process of determining how and in what measure the constituent parts of a project or a process, or sectors of an economy interact is enhanced by the application of IOA. When put into a matrix form, the rows represent the output of each sector to the other sectors. The columns show how each sector

obtains the inputs (i.e., goods and services) that it requires from the other sectors. The columns represent how a particular sector uses the inputs of the other sectors including inputs that it requires from its own industry. This matrix reflects the flow of trade between the different sectors of the economy. Table A.1 shows an example of the Leontief's I-O table with all units of measure given in dollars[1]. Thus, the table illustrates a grid market place with three grid resources[2] R_1, R_2, and R_3 and three grids G_1, G_2, and G_3. The columns represent the inputs (grid resources) to the grid marketplace and the rows represent the outputs (computations) from a grid. For example in Table A.1, column 2 on row 1 request for 19 units of R_2 from G_1, 7 units from G_2, and 10 units from G_3. The elements of Table A.1 show the flow relationship between three grid resources. The outputs (computations) show that G_1 produces 14 units of R_1, 19 from R_2, and 32 form R_3 to produce an output of 65.

Table A.1: Input-Output Table (in dollars).

		Inputs (Grid Resources)			Total Outputs ($)
		R_1	R_2	R_3	
Outputs (Grid Computations)	G_1	14	19	32	65
	G_2	10	7	15	32
	G_3	20	10	40	70
Total Outputs ($)		44	36	87	167

One important step in the analysis of the Leontief I-O table is the development

[1]For the purpose of illustration, we use the grid and an economy with grid resources as inputs and computations as the outputs

[2]The grid resources R_1, R_2, and R_3 can be any of the resources described in Chapter 1.

a Table of Technology Coefficients (TTCs). The TTCs provides the amount (unit) of input required from each input to generate one unit of the output for each grid computation. For example, in Table A.1, let the output elements be denoted by x_{ij} that is, computation j (G_1, G_2, G_3) that consumed resource i (R_1, R_2, R_3); and let X_i represent the total output from each grid, and y_i represents the final demand for each grid's output computation. The amount of resource R_2 from G_1 is given by $x_{G_1,R_1} = \$19$. Let a_{ij} be the value of the computed output that is required by the grid input resource i to produce one dollar's worth of output of grid computations j. The table of technology coefficients is obtained by a normalization, that is:

$$a_{ij} = \frac{x_{ij}}{X_j} \tag{A.1}$$

Using Table A.1, for example, TTC for a_{11} can be computed as $a_{11} = \frac{14}{65} = 0.215$, $a_{12} = \frac{19}{32} = 0.594$ and $a_{13} = \frac{32}{70} = 0.457$ continuing in this manner, the TTC for this specific example is computed and given in Table A.2. The computed TTCs shows

Table A.2: Computed Technology Coefficients.

		Inputs (Grid Resources)		
		A	B	C
Outputs (Grid Computations)	A	0.215	0.594	0.457
	B	0.154	0.219	0.214
	C	0.308	0.313	0.571

that the total amount required to compute one dollar's worth of the grid output for

resource A (for example), is given as:

$$\sum_{i=1}^{3} a_{i1} = 0.215 + 0.154 + 0.308 = 0.677. \tag{A.2}$$

Appendix B

Derivation of the Black-Scholes Formula

B.1 Prerequisites

In this appendix, I intend to provide the derivation of the Black-Scholes formula, hence, the enthusiastic reader is encouraged to note the following properties of the normal distribution and the mean value operator. Let X be normally distributed, that is $X \sim N(\mu, \sigma^2)$. Then

$$\begin{aligned} P(X \leq x) &= N(x) \\ &= \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) dy \end{aligned} \tag{B.1}$$

The function $N(x)$ is called the distributed function of X and the function

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) \tag{B.2}$$

is termed density function. Note that because of symmetry, $N(x) = 1 - N(-x)$. The mean value of the random variable X can be calculated as

$$E[Z] = \int_{-\infty}^{x} y\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)dy \tag{B.3}$$

If $g : \mathcal{R} \to \mathcal{R}$ is an integrable function, then the integral transformation theorem provides a way of calculatio=ng the mean value of $Z = f(X)$:

$$\begin{aligned} E[Z] &= E[f(X)] \\ &= \int_{-\infty}^{x} f(y)\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)dy \end{aligned} \tag{B.4}$$

Equation B.4 proves useful in the derivation of the Black-Scholes formula.

B.2 Formula Derivation

I assume that the stock price in a risk-neutral world moves according to a geometric motion, that is

$$dS(t) = S(t)rdt + S(t)bdz(t). \tag{B.5}$$

From Hull [21], it is known that $S(t)$ can be written in the more explicit form:

$$S(T) = S(0)\exp\left((r - \frac{b^2}{2})T + bz(T)\right). \tag{B.6}$$

Since $z(T) \sim N(0,T), S(T)$ could be interpreted as a random variable of the form $S(T) = Y = f(z(T))$ where

$$f(x) = S(0)\exp\left((r - \frac{b^2}{2})T + bx\right). \tag{B.7}$$

All I need to do in order to calculate the value of an option is to calculate the expected value of the pay-off and discount it using the risk-free rate r. Hence, for a European call option initiated at date 0 with an exercise price of K and T years to expiry I have to calculate.

$$\begin{aligned} c(0) &= \exp(-rT)E[max(S(T) - K, 0)] \\ &= \exp(-rT)E[(S(T) - K)^+]. \end{aligned} \tag{B.8}$$

Again, the random variable inside the expectation is just a more completed function of the Wiener process. Specifically

$$\begin{aligned} (S(T) - K)^+ &= \left(S(0)\exp((r - \tfrac{b^2}{2})T + b\sqrt{T}\tfrac{z(T)}{\sqrt{T}}) - K\right)^+ \\ &= g(\tfrac{z(T)}{\sqrt{T}}). \end{aligned} \tag{B.9}$$

Note that if $z(T) \sim N(0,T)$, then $\frac{z(T)}{\sqrt{T}} \sim N(0,1)$. So to calculate the mean value in Equation (B.8), I apply Equation (B.4) by putting $\mu = 0, \sigma^2 = 1$ and $X = \frac{Z(T)}{\sqrt{T}}$:

$$\begin{aligned} E[(S(T)-K)^+] &= E[g\left(\tfrac{z(T)}{\sqrt{T}}\right)] \\ &= \int_{\infty}^{x} g(y)\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy \\ &= \int_{-\infty}^{\infty}\left(S(0)\exp((r-\tfrac{b^2}{2})T+b\sqrt{T}y)-K\right)^+ \\ &\quad \tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy. \end{aligned} \tag{B.10}$$

This integral looks moderately unpleasant, but note that

$$\begin{aligned} g(y)\geq 0 &\Leftrightarrow S(0)\exp((r-\tfrac{b^2}{2})T+b\sqrt{T}y)\geq K \\ &\Leftrightarrow (r-\tfrac{b^2}{2})T+b\sqrt{T}y)\geq \ln\left(\tfrac{K}{S(0)}\right) \\ &\Leftrightarrow y\geq \frac{\ln\left(\frac{K}{S(0)}-(r-\frac{b^2}{2})\right)}{T}b\sqrt{T} \\ &\Leftrightarrow y\geq -\frac{\ln\left(\frac{K}{S(0)}+(r-\frac{b^2}{2})\right)}{T}b\sqrt{T} \\ &\underline{\underline{\Delta}} \quad -d_2. \end{aligned} \tag{B.11}$$

Since the integrand is zero whenever $y \geq -d_2$ I can replace the lower limit in the integral in Equation (B.10) by $-d_2$ so that

$$\begin{aligned} E[(S(T)-K)^+] &= \int_{-d_2}^{\infty}\left(S(0)\exp((r-\tfrac{b^2}{2})T+b\sqrt{T}y)-K\right)\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy \\ &= \int_{-d_2}^{\infty}\left(S(0)\exp((r-\tfrac{b^2}{2})T+b\sqrt{T}y)\right)\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy \\ &- \int_{-d_2}^{\infty}K\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy \\ &= (1)-(2) \end{aligned} \tag{B.12}$$

The last two integrals are calculated separately:

$$\begin{aligned}(1) &= \int_{-d_2}^{\infty}\left(S(0)\exp((r-\tfrac{b^2}{2})T+b\sqrt{Ty})\right)\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy\\ &= \int_{d_2}^{\infty}S(0)\tfrac{1}{\sqrt{2\pi}}\exp\left((r-\tfrac{b^2}{2})T+b\sqrt{T}y-\tfrac{y^2}{2}\right)dy\\ &- S(0)\exp(rT)\int_{-d_2}^{\infty}\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{b^2}{2}T+b\sqrt{T}y-\tfrac{y^2}{2}\right)dy\\ &= S(0)\exp(rT)\int_{-d_2}^{\infty}\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{1}{2}(y-b\sqrt{T})^2\right)dy.\end{aligned} \tag{B.13}$$

The last equality is an application of the formula $a^2+b^2-2ab=(a-b)^2$. Using integration by substitution, I can put $v=y-b\sqrt{T}$. Then the limits of the integral have to be changed: $y=-d_2\Rightarrow v=-d_2-b\sqrt{T}\approx -d_1, y=\infty\Rightarrow v=\infty$ therefore,

$$\begin{aligned}(1) &= S(0)\exp(rT)\int_{-d_1}^{\infty}\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{v^2}{2}\right)dv\\ &= S(0)\exp(rT)\left(\int_{-\infty}^{\infty}\exp\left(\tfrac{v^2}{2}\right)dv-\int_{\infty}^{-d_1}\tfrac{1}{\sqrt{2\pi}}exp\left(-\tfrac{v^2}{2}\right)dv\right)\\ &= S(0)\exp(rT)(1-N(-d_2))\\ &= S(0)\exp(rT)N(d_1).\end{aligned} \tag{B.14}$$

This follows since I recognize the integrals as integrals over the density function of a standard normally distributed random variable. If this observation is kept in mind, the calculation of (2) is easy

$$\begin{aligned}(2) &= \int_{-d_2}^{\infty}K\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy\\ &= K\int_{-d_2}^{\infty}\tfrac{1}{\sqrt{2\pi}}\exp\left(-\tfrac{y^2}{2}\right)dy\\ &= K(1-N(-d_2))\\ &= KN(d_2).\end{aligned} \tag{B.15}$$

I tidy up the calculations to make the call option value easily calculated by discount-

ing:

$$\begin{aligned} c(0) &= \exp(-rT)E[(S(T)-K)^+] \\ &= \exp(-rT)\{(1)-(2)\} \\ &= \exp(-rT)\{S(0)\exp(rT)N(d_1)-KN(d_2)\} \\ &= S(0)N(d_1)-K\exp(-rT)N(d_2). \end{aligned} \tag{B.16}$$

This concludes the derivation of the Black-Scholes formula.

Appendix C

Service Level Agreement (SLA) and Quality of Service (QoS)

Grid computing has continued to evolve and has become an infrastructure for providing computational intensive services in research and commercial environments. As the grid continue to gain wide acceptance, there is need for techniques that could provide the required service objectives. This could be facilitated by means of electronic contracts between service consumers and one or more service provider(s), in order to achieve the necessary reliability and commitment on both sides. Such contracts help to establish a well-defined relationship between a service provider and a client in the context of a particular service provision. This is especially important if the services or resources to be used come from different administrative domains, or if commercial service provision needs to be supported. Over the last years Service Level Agreements (SLA) increasingly been used to establishing these kinds of guarantees and relationships.

C.1 Service Level Agreement (SLA)

A Service Level Agreement (SLA) is defined as a contract between a provider and a user, which specifies the level of service that is expected during the term of the contract [7]. The distributed computing community's interest in SLA comes from the need to guarantee qualities of service (QoS) for distributed services and composite applications built from those services.

The SLA records a common understanding about services, priorities, responsibilities, guarantees, and warranties. Each area of service scope should have the "level of service" defined. The SLA may specify the levels of availability, serviceability, performance, operation, or other attributes of the service, such as billing. The "level of service" can also be specified as "target" and "minimum", which allows customers to be informed what to expect (the minimum), whilst providing a measurable (average) target value that shows the level of organization's performance. In some contracts, penalties may be agreed upon in the case of non-compliance of the SLA.

The characteristic nature of grid resources and their ownership however, hinder the grid from plugging directly into the default SLA that is generally defined. The reason is that in grid the SLA applicable to one (based on the geographical location and policies) may not necessarily apply to other grids. In order words, it is hard to define a global SLAs for grids. However, a SLA document may contain the following numerous service performance metrics with corresponding service level objectives. A common case with grid could be designed with the following metric:

1. ABA (Abandonment Rate): Percentage of jobs abandoned while waiting for service. This metric is sensitive and it must be ensured that it is kept as

minimum as possible since flexibility in terms of modeling with real option for the users is built into the pricing model.

2. ASR (Average Speed to Respond): Average time (usually in hours or minutes) it takes for a job to remain in queue before the required computational resources become available.

3. TSF (Time to Service Factor): Percentage of jobs served within a definite time frame, for example, 99% in 2 hours.

4. RA (Reservation Acknowledged): Percentage of priority jobs that can be resolved without the use of a incentive or issuing a penalty.

5. TAT (Turn Around Time): Time taken to complete a certain job.

Therefore, to draft an effective SLA for grid, the service levels (such as the following) that an existing infrastructure can support needs to be identified to make the SLA comprehensive:

1. The SLA need to clearly define the service provider's responsibilities (service provisioning).

2. There is a need to negotiate the SLA with the service provider, while paying particular attention to what services are being guaranteed, how they will be measured, the process for realizing agreed-upon remedies, and the amount of time the service provider has to correct problems if problems occurs.

3. There is a need to implement SLA measurement and enforcement tools and processes to ensure that every SLA can be measured and enforced as soon as

the service under consideration is provided.

4. There is a need to enforce SLA compliance, and identify and resolve problems that arise.

5. There is also, a need to tie the SLA document to government legislations regarding compliances from both parties (provider and user).

C.2 User Quality of Service (QoS)

A user's QoS refers to the users perception of a service to the set of predefined service conditions (service agreements) necessary to achieve the specified service quality as described in the Service Level Agreement document. $QoS = \{vLQoS, LQoS, MQoS, HQoS, vHQ$ is a fuzzy set of linguistics variables vLQoS, LQoS, MQoS, HQoS, and vHQoS used to denote very low QoS, Low QoS Mid QoS, high QoS, and very high QoS respectively. To compute the corresponding membership functions, the following rated values for the linguistics variables in Table C.1:

Table C.1: The Rated QoS Levels.

Linguistics Variables	a	b	c	d	Crisp Values (z^*)
$VLQoS$	a_1	b_1	c_1	d_1	$z^*_{v_1}$
$LQoS$	a_2	b_2	c_2	d_2	$z^*_{v_2}$
$MQoS$	a_3	b_3	c_3	d_3	$z^*_{v_3}$
$HQoS$	a_4	b_4	c_4	d_4	$z^*_{v_4}$
$VHQoS$	a_5	b_5	c_5	d_5	$z^*_{v_5}$

To defuzzify[1] I use Centre of Area (COA) method [28]. The COA method is suit-

[1]Defuzzification involves the process whereby a non-fuzzy (crisp) output B^* is obtained from the

able for point-wise membership functions such as triangular and trapezoidal membership functions. The COA method calculates the centre of gravity of the distribution of the degrees of membership of B^*. In a discrete universe U, the crisp output value z^* is obtained using Equation (C.1).

$$z^* = \frac{\sum_{i=1}^{q} B^*(x_i)\dot{x}_i}{\sum_{i=1}^{q} B^*(x_i)}. \tag{C.1}$$

where q is the number of quantization levels of the universe U, x_i is the crisp value for the quantization level i, and $B^*(x_i)$ is the inferred fuzzy set, B^*.

$$\mu QoS(a_1, b_1, c_1, d_1) = \begin{cases} 0 & t < a_1, \text{and } t \geq d_1 \\ 1 & b_1 \leq t < c_1 \\ \frac{t-a_1}{b_1-a_1} & a_1 \leq t < b_1 \\ \frac{d_1-t}{d_1-c_1} & c_1 < t \leq d_1 \end{cases} \tag{C.2}$$

Generally, the remaining individual membership function corresponding to the $QoS = \{vLQoS, LQoS, MQoS, HQoS, vHQoS\}$ can be computed using the following:

$$\mu LQoS(a_2, b_2, c_2, d_2) = \begin{cases} 0 & t < a_2, \text{and } t \geq d_2 \\ 1 & b_2 \leq t < c_2 \\ \frac{t-a_2}{b_2-a_2} & a_2 \leq t < b_2 \\ \frac{d_2-t}{d_2-c_2} & c_2 < t \leq d_2 \end{cases} \tag{C.3}$$

fuzzy set, which results from fuzzy inference.

$$\mu MQoS(a_3, b_3, c_3, d_3) = \begin{cases} 0 & t < a_3, \text{and } t \geq d_3 \\ 1 & b_3 \leq t < c_3 \\ \frac{t-a_3}{b_3-a_3} & a_3 \leq t < b_3 \\ \frac{d_3-t}{d_3-c_3} & c_3 < t \leq d_3 \end{cases} \tag{C.4}$$

$$\mu HQoS(a_4, b_4, c_4, d_4) = \begin{cases} 0 & t < a_4, \text{and } t \geq d_4 \\ 1 & b_4 \leq t < c_4 \\ \frac{t-a_4}{b_4-a_4} & a_4 \leq t < b_4 \\ \frac{d_4-t}{d_4-c_4} & c_4 < t \leq d_4 \end{cases} \tag{C.5}$$

$$\mu VHQoS(a_5, b_5, c_5, d_5) = \begin{cases} 0 & t < a_5, \text{and } t \geq d_5 \\ 1 & b_5 \leq t < c_5 \\ \frac{t-a_5}{b_5-a_5} & a_5 \leq t < b_5 \\ \frac{d_5-t}{d_5-c_5} & c_5 < t \leq d_5 \end{cases} \tag{C.6}$$

Now, QoS can be expressed as:

$$QoS = \{z^*_{v_1}, z^*_{v_2}, z^*_{v_3}, z^*_{v_4}, z^*_{v_5}\} \tag{C.7}$$

C.2.1 Mapping QoS to Computed Option Value

The RHS of Equation (C.7) is crisp value generally expressed as a percentage. An example for mapping a given user QoS to the computed option value is given below.

C.2.2 A Mapping Example

In Figure 6.10, the option value has (or convergence) occurred at $\$1.532 \times 10^{-4}$ during the 8th time step. However, I also have an option value that occurred during the 4th time step as $\$1.524 \times 10^{-4}$. This is my minimal option value for this particular gcc (RAM). Between these values $\$1.532 \times 10^{-4}$ (maximum) and $\$1.524 \times 10^{-4}$ (minimum) , users may have their jobs executed in the grid using the gccs. For example, users who offer to pay the minimum will be mapped to $z^*_{v_1}$ QoS while users who agree to offer the maximum option value will be mapped to $z^*_{v_5}$. Every other offer in the middle are correspondingly mapped. I also make a provision for users who want their jobs executed in front of every other users. That means they offer to pay more than the maximum of the computed option value of $\$1.532 \times 10^{-4}$. In this case, the extra pay is given in the form of incentive (by way of granting more resources to the bumped out user.)