Acknowledgements

We would like to seize this opportunity to personally thank all the GridEcon Consortium members who have been involved in the GridEcon Project from June 2006 - April 2009. Each member has brought a key skill and contribution that has been valuable to the research and completion of this Project, which would not have been possible without the funding from the European Union. Thank you.

Jörn Altmann and Sonja Klingert

Introduction

When the GridEcon project was launched in July 2006, the idea of Grid computing was mainly restricted to voluntary sharing of computing resources (e.g. supercomputing resources or distributed servers). The idea of commercial computing-on-demand was a vague vision that just started to show up in ICT business with Amazon's "Elastic Compute Cloud" (EC2) service, which became operational in its beta version on August 25th, 2006.

Throughout the duration of the GridEcon project, the commercial usage of Grid computing in the Grid world outside GridEcon has evolved substantially. Not only has the EC2 service become a cornerstone of commercial computing resource trading since fall 2008; there are also many other companies that provide Cloud services, such as Sun N1 Grid and IBM Blue Cloud. In other words, the idea of commercial Grid computing has been augmented by the vision of the Cloud.

GridEcon has taken part in this evolution as it has refined the original idea of creating a market for computing power with a computing-on-demand business model. The timing is good as especially in phases of economic slumps there is a compelling rationale for a market place that helps at balancing demand spikes through markets. Besides, idle computing power can be sold on the market, thereby reducing the operating costs of the datacenter. The GridEcon testbed comprises not only a marketplace where virtualized compute resources can be traded using two different market mechanisms. Beyond that it offers a comprehensive set of services which supports the buyer and seller in using the GridEcon testbed. The GridEcon economic analysis identified core services (e.g. monitoring service), value-added services (e.g. capacity planning) and infrastructure support services (e.g. workflow engine).

The achievements of GridEcon have been published in many publications. A selection of those published and peer-reviewed papers has been compiled into this "GridEcon book".

The first part of this compendium of GridEcon-related papers starts with an introduction to GridEcon issues, using papers of Jörn Altmann, Costas Courcoubetis. John Darlington, and Jeremy Cohen, which were published in the GECON workshops (Grid Economics and Business Models) 2007 and GECON 2008, both organized by the GridEcon consortium. These papers are completed by a paper titled "Taxonomy of Grid Business Models" by Jörn Altmann, Ashraf A. Bany Mohammed and Mihaela Ion, that evaluates the business models that can be applied to use the Grid successfully. The basic ideas about business aspects of a Grid marketplace can be found in "Adopting the Grid for Business Purposes: The Main Objectives and the Associated Economic Issues" by George A. Thanos, Costas Courcoubetis, and George D. Stamoulis.

The second part of our "GridEcon Book" deals with the economics of GridEcon. Within the basic GridEcon economic concepts, one major optimization problem is the auction-based resource allocation, which is addressed in the paper of Manos Dramitinos, George D. Stamoulis, and Costas Courcoubetis. Issues around the problem of "Pricing resources on demand" by Costas

Courcoubetis, Sergios Soursos and R.R. Weber that formed one of the intellectual starting points of the project have been at the core of the economics of GridEcon from the very beginning. The temporary status of this discussion can be found in “Dynamic bandwidth pricing: provision cost, market size, effective bandwidths and price games” by Sergios Soursos, Costas Courcoubetis, and R. Weber.

Putting together all these separate issues leads to the “Cost Analysis of Current Grids and its Implications for Future Grid Markets” by Marcel Risch and Jörn Altmann which gives an analysis with regards to the cost-effectiveness of Grid resources, using the Amazon EC2 services as a starting point.

The subject of the last section is the technical side of GridEcon: How to realize economics-aware components in a Grid market. A fundamental problem that had to be solved was the mapping of workflows onto Grid resources in the context of SLAs. This issue is discussed in the papers of Dang Minh Quan and Jörn Altmann. One of the value-added Grid marketplace services that was judged to be vital for resource buyers with little IT expertise, was the capacity planner. Issues around the design of this component are dealt with in “Economics-Aware Capacity Planning for Commercial Grids” by Marcel Risch, Jörn Altmann, Yannis Makrypoulias, and Sergios Soursos in “Collaborations and the Knowledge Economy”. Publications on the implementation and the GridEcon Platform are in preparation, and are planned to be submitted to major conferences and journals.

After nearly three years of intense work on developing a platform for real world marketplaces for computing power, it remains to be asked is: What will be the legacy of the GridEcon project for the future of the Grid? One evolution of the Grid market will most probably be clouds where users post bids and asks for tasks that they would like to purchase or offer. Such a service will rely heavily on the research done by GridEcon on topics such as pricing, market components, good definitions, and market mechanisms. Yet another evolution of the Grid might be the virtual merging of computing powers of virtually connected companies to form an intra-company Grid, balancing different needs for computing power within a common project. This kind of organization will also need the findings of GridEcon in order to assign resources efficiently.

Whichever direction distributed computing is taking, chances are big, that GridEcon results will be incorporated into solutions if a market approach is involved.

Contents Page

I. Commencement of GridEcon: Economic Issues in Grid Research

I.i. Jörn Altmann, Costas Courboubetis, John Darlington, Jeremy Cohen "GridEcon - The Economic-Enhanced Next-Generation Internet", GECON 2007, LNCS 4685, Springer 2007;

I.ii. GridEcon: A Market Place for Computing Resources Jörn Altmann, Costas Courcoubetis, George D. Stamoulis, Manos Dramitinos, Thierry Rayna, Marcel Risch, Chris Bannink, GECON 2008, LNCS 5206, Springer 2008

I.iii. Jörn Altmann, Ashraf A Bany Mohammed, and Mihaela Ion, "Taxonomy of Grid Business Models", Proceedings of Grid Economics and Business Models, Rennes, France, August 2007

II. The Economics of GridEcon

II.i. C Courcoubetis, S Soursos and RR Weber, "Pricing Resources on Demand", 1st IEEE International Workshop on Bandwidth on Demand (BoD 2006), In conjunction with IEEE Globecom 2006, San Francisco, California;

II.ii. George A. Thanos, Costas Courcoubetis and George D. Stamoulis, "Adopting the Grid for Business Purposes: The Main Objectives and the Associated Economic Issues", The 4th International Workshop on Grid Economics and Business Models, GECON 2007.

II.iii. S. Soursos, C. Courcoubetis and R. Weber, "Dynamic bandwidth pricing: provision cost, market size, effective bandwidths and price games", Journal of Universal Computer Science (J.UCS.), Special issue on Socio-Economic Aspects of Next Generation Internet, Spring 2008

II.iv. Risch, M., Altmann, J.: Cost Analysis of Current Grids and its Implications for Future Grid Markets. In: Proceedings of the Grid Economics and Business Model Workshop. Gecon 2008, pp.13-27. Springer LNCS. Heidelberg, 2008.

III. GridEcon Technical Solutions: Economics-Aware Components in a Grid Market

III.i. Minh Dang Quan, "Mapping heavy Communication Workflows onto Grid Resources within SLA Context", Proceedings of the Second International Conference on High Performance Computing and Communications (HPCC-06), Munich, Germany, September 2006;

III.ii. Dang Minh Quan and Jörn Altman, "Mapping a Group of Jobs in the Error Recovery of the Grid-based Workflow within SLA Context", Proceedings of the 21st International Conference on Advanced Networking and Computing, Niagara Falls, Canada, May 2007;

III.iii. D.M. Quan and J. Altmann "Mapping of SLA-based workflows with light communication onto Grid resources.", to be published in the Proceeding of the 4th International Conference on Grid Service Engineering and Management - GSEM 2007;

III.iv. Risch, M., Altmann, J., Makrypoulias, Y., Soursos, S.: Economics-Aware Capacity Planning for Commercial Grids. In: Collaborations and the Knowledge Economy, Volume 5. pp. 1197-1205. IOS Press. Amsterdam, 2008.

GridEcon – The Economic-Enhanced Next-Generation Internet

Jörn Altmann[1,2], Costas Courcoubetis[3], John Darlington[4], Jeremy Cohen[4]

[1]Intl. University, Bruchsal, Germany
[2]Seoul National University, Seoul, South-Korea
[3]Athens University of Economics and Business, Athens, Greece
[4]Imperial College, London, UK
jorn.altmann@acm.org; courcou@aueb.gr; jd@imperial.ac.uk

Abstract. The major shortcoming of Grid middleware systems is the lack of economic-enhanced Grid services. These new services are necessary in order to let Grid users benefit from the properties of the Grid. Those properties comprise the availability of on-demand computational power, simplicity of access to resources, low cost of ownership, and a pay-for-use pricing model in addition to the already leveraged properties such as cost reduction and aggregated processing power for high-performance computing applications. This paper gives an overview of the EU-funded project GridEcon on Grid economics and business models. It describes its vision of the next generation Grid/Internet, in which individuals, universities, small and medium sized enterprises (SMEs), and large companies have access to the Grid in exactly the same way. Any resource, including servers, storage, software, or data, is accessible as a service. In addition to this, the architecture of an economic-enhanced infrastructure is illustrated and the goal of the project is described.

Keywords: Grid Computing, Grid Economics, Service-Oriented Computing, Economic Modeling, Business Model, Markets, Architecture, and Next-Generation Internet.

1 Introduction

Grid computing has not been commercially taken up to the extent expected during the past few years, although many different (commercial and public domain) Grid middlewares (e.g. glite, Gria, Unicore, Globus, GridBus) have been designed and developed [1][2][3][4][5]. The reason is hidden in the limited leverage of the properties of Grid technology. Currently, enterprises use Grid technology only to consolidate their IT resources, resulting in cost reduction. Only in a few cases, Grid technology is being used for improving the workflow within an enterprise. For example, the combined processing power of geographically distributed servers can be used to reduce the processing time of calculations, or to calculate equations more accurately. It results in reduced time-to-market of products. Grid technology also helps aggregating high-performance computing resources such that applications,

generating more precise results, can be executed on those aggregated resources. The execution of these applications on a single high-performance computer would not work.

However, enterprises miss out on using other properties of Grid technology. These properties comprise the availability of on-demand computational power, simplicity of access to resources, low cost of ownership, and a pay-for-use pricing model. On-demand computational power helps enterprises to deal with unexpected demand economically efficient. Instead of declining a consumer's request simply based on the unavailability of resources (i.e. processing power, storage, bandwidth, software, and data), they could buy those resources on the Grid (if it maximizes the enterprises objective) now. The simplicity of access to resources helps users to access any resource without much effort. Low cost of ownership enables small and medium sized enterprises (SMEs) to get access to resources that they could not afford to purchase as a whole. They only have to pay for the usage of the resources. This model would allow them to compete with large companies, which have the financial resources to buy high-performance computers for their applications.

Considering this situation, two questions arise: First, what is the reason for this low take up of Grid technology; Second, are there no further sustainable business models then those three mentioned above? These questions highlight the need for better understanding the economics behind Grid technology as well as their business models. The GridEcon project addresses these questions [7]. The GridEcon project investigates the economics of participation in a Grid environment as well as how economic principles can be integrated into existing Grid middleware to make it economic-aware. Current Grid middleware lacks these capabilities, as has been analyzed in [8]. A taxonomy of business models has been proposed in [6].

2 Vision of the Future Grid

In a future Grid, which we envision to be the next-generation Internet (i.e Web 3.0), an open market (together with its trading system) is an essential part, where a huge variety of electronic services are traded. Participants (both, consumers and providers) in this market could be anyone from the general public, academia, business, and government, making it a rich economic and social environment. Based on these markets, sustainable Grid business models could be created, offering new ways to generate income. The income could come from customization of information or the creation of new workflows. These new business models would allow participants in the Grid economy to buy services and sell enhanced services at the same time [11].

However, this vision has not been implemented yet. The reason can be found in the fact that there is one technology out of four that is still missing. All of them are necessary to make the vision become reality. The three existing technologies are: service-oriented computing, virtualization of resources, and network computing. Service-oriented computing (e.g. Web services) allows useful capabilities to be encapsulated as easy-to-use, composable services. Hardware virtualization technology allows transparent use of distributed resources. Network computing allows uniform access to the Internet, which is enabled through the convergence of networks and the

proliferation of broadband access. The only missing technology is economic-enhanced services, which will give participants in the market tools to evaluate the economic risk and opportunity to engage in a transaction.

This technology will have a significant impact on existing Grid businesses such as location-aware mobile services, consumer advice services, utility computing, brokers, virtual facilities, insurance contracts, software-as-a-service, and information-as-a-service. It will make them accessible to a larger base of customers.

3 Architecture

Looking at the currently available Grid middleware solutions, it becomes obvious that all of the existing Grid middleware solutions do not provide economic-enhanced Grid services. To rectify this situation, the functionality of Grid technology must be enhanced so that an economic-aware operation of Grid services becomes possible. This new functionality would reduce uncertainty and give incentives to end-users not only to consume but also to sell services on the Grid. It could also help stakeholders to resolve their conflicts in preferences. It would, thus, create a new economy, in which all stakeholders can actively participate. An abstract view of this next-generation architecture is shown in the following figure.

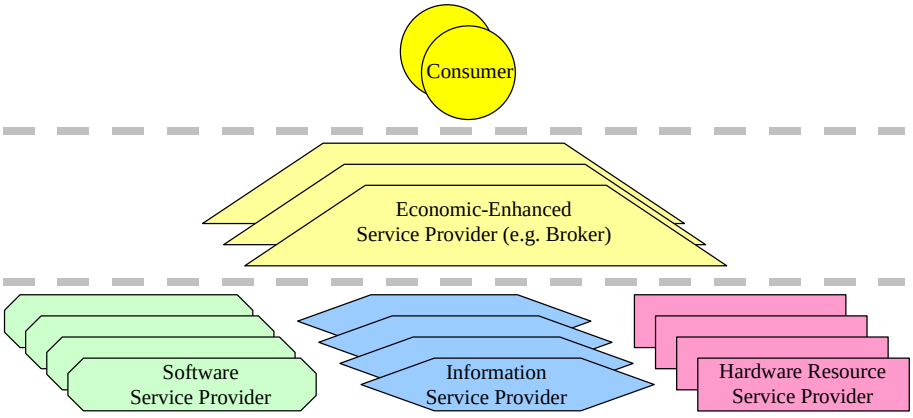


Fig. 1. Architecture of the economic-enhanced next-generation Grid

The Economic-Enhanced Service Provider of Figure 1 will provide tools for trading resources (i.e. software, information, and hardware resources) [10]. It will help Grid stakeholders (i.e. researchers, organizations, companies, and the general public) to deal with the currently existing shortcomings of Grid computing such as risk of relying on outside-company resources, lack of trust, risk in commitment to resource purchases, and uncertainty in capacity planning. These tools, which still need to be developed, range from risk broker services, capacity planning services, to services markets. The risk broker would offer a type of insurance contract to protect against financial loss from unavailable Grid resources or failed Grid resources. An

accurate capacity planning tool, which is vital for service provider and end-users, would give support for making decisions about when to purchase new servers, when to put spare resources on the Grid market, and when to buy resources from the Grid. The software services market would allow trading of units of software access. The price of the software access unit would include the price for the software usage and the charge for the hardware resources on which the software would be executed [9]. A hardware resource market will allow selling different server units under a specific pricing scheme. The following figure shows a few examples of hardware resource markets and their relationship to software markets.

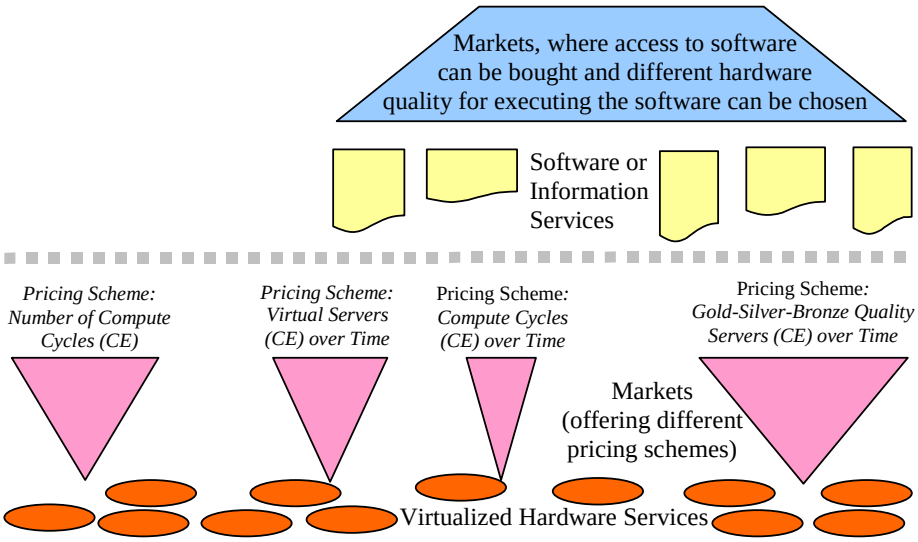


Fig. 2. Architecture of software, information, and hardware resource markets

These kinds of markets, as shown in Figure 2, are the basic services that are needed to make the Grid economic-enhanced. On top of those market services, services as the one mentioned above can be constructed.

There are two major threats. The first threat to Grid computing is the failure of developing and deploying those economic-enhanced Grid services. Since those services also require an open, economic-enhanced architecture for Grid services, which allows any stakeholder to plug in its own services, the second threat to Grid computing is that the Grid community fails to define such an open Grid services architecture.

A market that is based on this architecture will enable collaboration across individual organizational boundaries and reduce the participation risk of Grid stakeholders by allowing economically fair sharing of costs and generated value.

4 Goals of GridEcon

The goals of GridEcon are to twofold. On the one hand, the project has to identify missing technology and software. This comprises the design of the required economic enhancements to Grid technology (as described in the previous section), the implementation of a subset of these service enhancements, as well as the simulation of the workings of the enhancements. On the other hand, the GridEcon project will perform economic and business modeling. It will develop models showing how hardware, software, and information services can be bought and sold on the Grid. It will also investigate potential ecosystems and explore current and future business models.

In particular, the goals of GridEcon are to address the following issues: specify user requiments for accessing economic-enhanced services; SLA composition with respect to pricing; consumer, provider, and service reputation management; service API specification; future and spot markets, insurance contracts, and reservation schemes.

5 Conclusions

This paper discussed the opportunities that come with Grid computing. In particular, it presented the vision of the GridEcon project and the architecture of the future Grid, i.e. the next-generation Internet. The architecture comprises three layers of stakeholders: the basic resource providers (hardware, software, and information); the economic-enhanced service providers; and the consumers. We also showed how markets are the basic building block for other economic-enhanced services in the layer of the economic-enhanced service provider. Finally, we illustrated the different working areas of the GridEcon project and the challenges in this area of research.

Acknowledgement

This work has partially been funded by the European Union in the IST programme “Advance Grid Technology, Systems, and Services” under grant FP6-2005-IST5-033634 with the title GridEcon – Grid Economics and Business Models.

References

1. glite homepage, http://glite.web.cern.ch/glite, June 2007.
2. Gria homepage, http://www.gria.org, June 2007.
3. Unicore homepage, http://www.unicore.eu, June 2007.
4. Globus homepage, http://www.globus.org, June 2007.
5. GridBus homepage, http://www.gridbus.org, June 2007.

6. J. Altmann, M. Ion, and A. Mohammed, “A Taxonomy of Grid Business Models,” Gecon2007, Intl. Workshop on Grid Economics and Business Models, Rennes, France, August 2007.
7. GridEcon, Grid Economics and Business Models, EU funded project, http://www.gridecon.eu, June 2007.
8. A. Caracas and J. Altmann, “A Pricing Service for Grid Computing,” at http://it.i-u.de/schools/altmann/?page_id=26, 2007.
9. J. Darlington, J. Cohen, and W. Lee, “An Architecture for a Next-Generation Internet Based on Web Services and Utility Computing,” ETNGRID2006, Third International Workshop on Emerging Technologies for Next-generation Grid, University of Manchester, Manchester, UK, June 2006.
10. J. Cohen, W. Lee, J. Darlington, and A. S. McGough, “A Service-Oriented Utility Grid Architecture Utilising Pay-per-Use Resources,” COMSWARE2006, Utility Grids 2006 Workshop, 1st International Conference on Communication System Software and Middleware, Delhi, India, January 2006.
11. J. Cohen, J. Darlington, and W. Lee, “Payment and Negotiation for the Next Generation Grid and Web,” Concurrency and Computation: Practice and Experience. Online ISSN: 1532-0634, Print ISSN: 1532-0626, John Wiley & Sons, Ltd. Published Online, May 2007.

GridEcon: A Market Place for Computing Resources

Jörn Altmann[1,5], Costas Courcoubetis[2], George D. Stamoulis[2], Manos Dramitinos[2], Thierry Rayna[4], Marcel Risch[1], Chris Bannink[3],

[1] International University in Germany, School of Information Technology
Campus 3, 76646 Bruchsal, Germany
jorn.altmann@acm.org, marcel.risch@i-u.de
[2] Network Economics and Services Group (N.E.S.), Department of Informatics,
Athens University of Economics and Business (AUEB),
76 Patision Street, Athens, GR 10434, Greece
{courcou, mdramit, gstamoul}@aueb.gr
[3]Logica Management Consulting
Prof. W.H. Keesomlaan 14, 1180 AD Amstelveen, The Netherlands
chris.bannink@logica.com
[4]Internet Centre, Imperial College London
South Kensington Campus, 180 Queen's Gate, London SW7 2AZ, UK
t.rayna@imperial.ac.uk
[5] TEMEP, School of Engineering, Seoul National University
San56-1, Sillim-Dong, Gwanak-Gu, Seoul, South-Korea
jorn.altmann@acm.org

Abstract: This paper discusses the rationales for a Grid market and, in particular, the introduction of a market place for trading commoditized computing resources. The market place proposed makes computing resources from different providers substitutable through virtualization. This includes the definition of a spot and future market as well as the parameters that a market mechanism for computing resources should consider. The above market place is complemented by a set of value-added services (e.g. insurance against resource failures, capacity planning, resource quality assurance, stable price offering) that ensure quality for Grid users over time. The market place technology for all of the above services has been designed by the GridEcon project, contributing to a broader adoption of Grid technology and enabling a service-oriented knowledge utility environment.

Keywords: Commercial Grids, Grid Computing, Business Models, Grid Economics, Utility Computing, Market Mechanisms, Grid Market Place.

1 Introduction

The Grid, as we use it in this paper, is a system of interconnected, virtualized computing resources. Those computing resources can be located in a few data centers around the world (owned by large enterprises) or can be highly distributed (owned by many end-users or small and medium-sized enterprises). These virtualized hardware resources provide interfaces to execute software (e.g. applications or middleware).

There are many vendors that offer different kind of hardware virtualization software [14][15]. After the buyer purchased the right to execute the resources, it can upload an application or, first, a Grid middleware application and, then, an application on top of it. In our definition of the Grid, we explicitly exclude software resources or information resources.

During the past years, a few approaches have been undertaken to offer Grid technology for commercial purposes: The most successful approach is the one of Amazon, namely Amazon EC2 [1]. Besides this utility computing approach, there are three hardware vendors (HP [2], IBM [3], SUN [4]) and one more Internet company entered the market (Google [5]) that provides similar utility computing services. Since only these few players are in the market, the market structure for utility computing is an oligopoly.

In order to break the oligopoly market structure, GridEcon, an EC-funded project [10][11], offers market place technology that allows many (small) providers to offer their resources for sale. The effect of a market place for computing resources can be illustrated with the following example. If computing resources are scarce due to low supply, the market price for computing resources will be high. Enterprises requiring resources during high price periods will invest in additional equipment. These additional resources can then be externalized when they are not needed, thereby increasing the overall capacity available on the market. The income generated by selling resources in the Grid market will act as an incentive to sell spare capacity if a market place is available.

The GridEcon project designs the technology that is needed to create an efficient market place for trading commoditized computing resources. The market place allows every owner of computing resources to offer spare computing resources as a standardized virtual machine. The challenge is to design this standardized virtual machine, which can be traded on the market place easily and allows establishing a competitive market. The market mechanism used has been designed to be simple for participants to use, and also economically sound. The later is concerned with inducing the right economic incentives to participants and avoiding unwanted strategic behavior leading to market dominance by large players. The GridEcon project also designed a series of value-added services on top of the market place (e.g. insurance against resource failures, capacity planning, resource quality assurance, stable price offering), ensuring quality of the traded goods for Grid users. The market place technology and the value-added services contribute to a broader adoption of Grid technology and enable a service-oriented knowledge utility environment.

This paper is structured as follows: Section 2 describes the rational for a market place for computing resources. In Section 3, the requirements for a spot and future market are discussed in detail. This includes the definition of a spot and future market as well as the prerequisites for a market mechanism for computing resources. Section 4 concludes the paper.

2 Markets and Market Places for Computing Resources

In general, a market describes the entirety of all transactions between a buyer and a seller in all forms. This includes direct transactions between a buyer and a resource provider as well as the transactions performed using the market mechanism provided within the market place. A market for computing resources comes into existence if there is discontinuous demand and redundancy of computing resources.

A buyer of a (virtual) computing resource in our Grid market is an entity that purchases the right to execute an application on a computing resource. Similarly, a seller of a computing resource sells the right of using a computing resource for a certain period of time.

A market place is an environment, which supports buyers and sellers in carrying out their trading transactions between each other [13]. The rules of interaction between the players in the market place are set through the market mechanism. The market place helps finding trading partners more easily by storing information centrally and by offering procedures to facilitate the matching between supply and demand. It also makes sure that fraudulent transactions do not happen.

2.1 Rationale for a Market for Computing Resources

In general, a market for utility computing will only work if one of the following conditions are met: (1) the pattern of individual demand for resources show spikes; (2) the units of computing power that are needed are smaller than the purchase of a computer could provide. In addition to this, all of the following conditions exist: (3) adequate technology for implementing utility computing (e.g. definitions of standardized interfaces); and (4) none-constraining regulations.

The demand spikes, which have been mentioned in condition (1), could be a consequence of the type of business that the company performs. The business itself might bring uncertainty about the need for computing resources. An example for such a business could be a company that creates animated movies. When a movie has to be rendered, the demand for computing resources is very high, otherwise very low. To cover this uncertainty, until now, companies had to over-provision their IT resources, which is expensive.

If condition (2) applies, then, without utility computing, most users would not be able to afford computing resources because of the high cost of ownership and their sporadic usage patterns. Utility computing allows them to get access to a very large quantity of computing resources for a short time on a per-usage basis.

Condition (3) is a necessity in order to be able to substitute a computing resource of one provider with a similar computing resource of another provider. Without this condition, the effort to connect to another provider might be too high. In order to efficiently use utility computing, technology must be available that helps the provider to organize its tremendous amount of computing resources in an efficient way. It must be guaranteed that the management cost for using the Grid is not higher than administrating resources that are owned.

Government regulations mentioned in condition (4) can have a huge impact on the organization and efficiency of providing computing resources. These regulation issues

must address areas such as data storage location, taxation, and access rights to computing resources. In order to make utility computing successful, a prerequisite is a set of supporting regulations.

If a market for computing resources is not available, an alternative solution is over-provisioning. That is, a user should have a permanent computing capacity that can meet the demand peaks, which in light of condition (1) are much higher than the average. However, besides the fact that this would defeat the purpose of utility computing, this is beyond the budget of many users and it causes economic waste of resource. Another alternative is not to meet all demand for processing resources, which consequently leads to missing an opportunity for getting additional revenues.

2.2 Economic Implications of a Grid Market Place

The rationale for a market place for computing goods is the current utility computing market. It is an oligopoly. The advantage of the few providers of utility computing (e.g. Amazon, HP, IBM, Google, Sun) in this oligopoly is that they have brand recognition and are trusted entities. However, these few providers offer their resources at a price higher than in a competitive market structure.

A market place provides an alternative to the existing oligopoly of utility computing providers in the market. If buyers and sellers accept (i.e. trust) the market place for executing their trades, it will increase the supply of computing resources in the market. Consequently, It will lower the price for buying computing resources in the market. Computing resources become even affordable to enterprises with low budget. All quality of service issues would be resolved by the market place. Similar to the case of stock exchanges, there could be more than one market place for trading computing resources.

The market place that we envision in GridEcon is an environment that allows SMEs to trade their resources. However, there may be also larger companies that will benefit from such a market place and its services [12]. For instance, it might be that large companies offer their spare capacity at the market place.

Eventually, we expect to see new business models arise. These business models will make entities act as brokers of computing resources to other companies, or offer other value-added services, complementing the market services.

2.3 Services for the Market Place for Computing Resources

In order to attract customers to the market place and get the market place concept accepted by users, the market place must offer a set of services that makes the use of the market place service convenient, secure, and less risky. This is the focus of the GridEcon project. The services, which we identified to be necessary for a market place environment, can be classified into core services and value-added services. The market place provider offers the market mechanism service and additional core services. These services are described in detail within the next subsection. The value-added services do not have to be offered by the market place provider but can be offered by independent service providers instead. The capacity planning service and

the insurance service are briefly described in subsection 2.3.2. We are convinced, only if these services are present, a market place for commoditized computing resources will work.

2.3.1. Core Services of the Market Place for Computing Resources

Resource Redundancy

The market place might provide resource redundancy in order to achieve service reliability even if a resource provider dishonors his commitment. The market place might also provide extra resources in order to increase the probability of a liquid market in times when demand is not matched by supply. In these cases, the market place deals with the risk of resource unavailability and will ease the bootstrap of a new market place.

Monitoring of Computing Resource Offers

In order to assure quality of the good offered (i.e. to assure that customers are truthful in declaring their computing resource postings), the market place provider may probe randomly the offered, not leased computing resources by running benchmark programs on them. In case that the computing resources have been sold on the market place, the market place requests ratings from the buyer of the resources, using a reputation system. The information within the reputation system is private to the market place. It will be used to decide whether to allow resource providers to sell goods in the market place in the future.

Security

The market place has to provide a secure environment. All communications among the market participants and the market place has to be encrypted. The market place also has to ensure that no viruses are spread between machines that are traded on the market place. It also builds in protection mechanisms that blocks buyers from getting access to resources of the provider beyond the border of purchased resources.

Simplicity

The market place has to enable access to computing resources in a transparent and simple way, using an intuitive user interface. Any transaction on the market place has to be simple, including the integration of the resource into the existing IT infrastructure of the buyer.

Anonymity

The market place has to ensure anonymity of sellers and buyers. This service is necessary in order to hide the identity of large providers/sellers. If the identity cannot be hidden, buyers and sellers might circumvent the market place and make the market transaction directly. If anonymity exists, buyers and sellers cannot trade directly with each other and more competition is guaranteed on the market place. However, buyers must be given the option of bidding for resources that will be provided by a single provider.

Standardization of Computing Resources

In order to offer commoditized computing resources, the market place must be able to cope with different hardware types available at the sellers' premises. Therefore, the market place requires sellers to virtualize their hardware and to run "standardized" virtual machines with certain performance characteristics (as defined by the market place provider). The market place accepts only offers of computing resources that comply with those performance standards. This makes all hardware resources comparable and substitutable.

The standardized resources are classified in terms of quality (e.g. CPU speed, bandwidth, main memory, disk space). However, in order to abstract from the detailed specification of the performance characteristics of virtual machines, those "standardized" Grid resources are given abstract names such as GEUnit1, GEUnit2, or GEUnit3.

2.3.2. Value-Added Services to the Market Place

Capacity Planning

The acceptance of Grid computing also depends on how simple is it to make optimized planning decisions about computing resource purchases. Therefore, to achieve acceptance, a capacity planning service needs to be in place that supports market place participants (i.e. sellers and buyers) in their decision making process for selling and buying resources on the market place. The capacity planning service has also to help customers on how to optimally shape their demand and to find the appropriate resources for their applications. The precision of the prediction of the capacity planner is based on input parameters, such as the current load, the past load, the current demand, the market price of computing resources, and the existing computing capacity.

Insurance Contracts

Uncertainty about resource failures can also have an impact on the acceptance of the market place. Those market participants, who are risk averse, will not participate in the market place if there is uncertainty about resource reliability. To overcome this, an insurance service must be in place. The insurance service provides an insurance contract to buyers for occurrences of resource failures. In case of a failure of a resource, the insurance provider replaces the failed resource with a fault-free resource (in case it owns resources) or simply compensates the buyer with the amount of money specified in the insurance contract.

3 Requirements for a Spot and Future Trading

Stocks are traded in markets called stock exchanges (e.g. New York Stock Exchange). Though all exchanges used to require physical presence of traders and trading was performed by means of open outcry, most modern stock markets rely on

automated electronic trading systems. For instance, NASDAQ is an electronic stock trading platform, where all trading is done by means of computer systems [6].

3.1 Definition of a Spot Market

In order to set the requirements for a spot market for computing resources, a general definition of spot markets and an example are given. In general, the spot market is a securities market, in which goods, both perishable and non-perishable, are sold for cash and delivered immediately or within a short period of time. Contracts sold on a spot market are also effective immediately. The spot market is also known as the “cash market” or “physical market.” Purchases are settled in cash at the price set by the market, as opposed to the price at the time of delivery. An example of a spot market commodity that is regularly sold is crude oil. Crude Oil is sold at the current prices, and physically delivered later [7].

The emergence of electricity wholesale markets is the consequence of privatization of the electrical power production companies. Like computation service, electricity is difficult to store (in large quantities), has to be available on demand, and (unpredictable) demand spikes may occur. Countries that have chosen to operate wholesale electricity markets where power companies offer their electricity output to meet the customers demand, have a number of mechanism to choose from. One model, which is used by the PJM [16] uses central scheduler to balance supply and demand and computes the market price, while the losses over the transmission network are also taken into account: At each network node a "shadow price" is computed, which reflects the cost of providing an additional MWatt-hour at this node.

Another model is that of conducting auctions in various time scales, i.e. auctions for yearly and daily provision of power, with additional spot market that resolves the need for accommodating short-term demand spikes. For example, this model is used by the European Energy Exchange (EEX) [17].

Finally, the Supply Function Equilibria has also been under investigation as the market mechanism of the power grids. It is also worth noting that there are several cases where regulators have intervened due to market failure, with the California market being the most prominent example [8][9].

3.2 Spot Market for Computing Resources

The spot market for computing resources also enables the trading of computing resources “as soon as possible”. It employs a bid and ask mechanism (i.e. a stock market-like double auction mechanism) that enables the trading of computational power. The underlying principle for this mechanism is that of a standard spot market: All parties publicly announce the maximum price they are willing to buy for and the minimum price they are willing to sell for. The spot bids (respectively asks) are put in the spot queue for bids (respectively asks). Matters are more complicated for our system’s spot market than in standard spot markets of storable commodities, since this good is non-storable and that the resource provisioning has to be transparent to the buyer.

3.3 Future Market for Computing Resources

The futures market for computing resources is actually a directory service containing the offers (respectively requests) for resources that are made available (respectively demanded) in a certain time interval. This index of offers and requests is searchable and visible to both bidders and providers. This market for futures complements the spot market. The futures and derivatives are contracts that denote the obligation of a buyer (respectively seller) to buy (respectively sell) at a certain agreed price.

3.4 Requirements for a Market Mechanism for Trading Computing Resources

After introducing briefly the spot and futures market for computing resources, we proceed to provide additional details regarding the unit that is to be traded in these markets (i.e. unit of trade), the format of spot market bids/asks and the futures market requests/offers, the matching algorithm to be adopted, and, finally, how bids and asks are routed in this system.

3.4.1. Unit of Trade

Prior to proceeding with the presentation of the GridEcon market mechanism, we define the unit that is to be traded in the Grid market place. Obviously, the unit must be suitable for the types of Grid applications currently existing or emerging. Computing resource providers offer different types of virtual machines (VMs) for leasing. It is expected that these resources be offered for a minimum desirable price and for a certain time duration within a specific time interval, depending on the providers' supply constraints. An assumption of our model for both the spot and the futures market throughout the paper is that time is discretized in time slots. Note that a virtual machine does not just correspond to a certain computational speed but rather to an entire configuration. The unit of trade, i.e. the VM, is defined through the CPU speed, the size of the main memory, and the size of the harddrive.

Depending on the nature of the tasks that consumers may wish to execute, their demand can be expressed in a multitude of ways. A general type of contract is specified by means of the number of VMs and the time duration. For instance, a company's Web server leases Grid resources when it is critically loaded. This type of consumer need can also be graphically depicted by means of a rectangle (see Fig. 1).

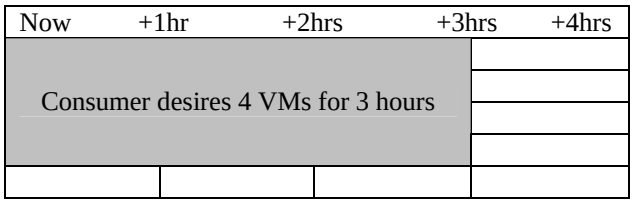


Fig. 1. A consumer's demand for 4 VMs over 3 hours is depicted as a rectangle.

The height of the rectangle denotes the number of virtual machines required at any time of the interval, while the width of the rectangle denotes the amount of time for which these machines are needed.

Another type of contract could be specified by means of computational volume, i.e. a total number of VMs must be made available up to a maximum deadline constraint, so that a certain computationally intensive task can be executed in time. As opposed to the previous case, only the total quantity of computational power is of interest, while the rate of computation provided at the various time epochs is not. This could be the case for data parallel applications. In this case, the consumer needs do no longer correspond to rectangles but rather to areas of rectangles, possibly with a maximum width constraint (i.e. deadline). Since this type of contract can be also expressed (with some effort on the consumer side) in the market place through the earlier type of contract, we will focus on the earlier contract type.

3.4.2. Format of the Bid/Ask

A bid in our system describes the resources required by the buyer. The resources are specified according to:

(1) The type of resource (VM) required,
(2) The quantity of resources (the number of VMs) required,
(3) The start time of the interval for using the resources (VM),
(4) The time duration of using the resources,
(5) The price expressed in €/min/unit, and
(6) The time limit until which the bid is valid. If the time limit is reached without the bid being matched, the bid is removed from the system.

In order to keep the definition of the bid as general and flexible as possible, instead of allowing only fixed values for the number of VMs and the time duration, we allow that the bid can specify whether these constraint values should be met with equality or $\leq$ or $\geq$. Our system also considers two additional constraints regarding the total expenditure and the total volume of computation (see the "Rectangle" column in Table 1). Therefore, since each bid is associated with a set of relation constraints, this allows a richer ontology of bids. A meaningful subset of this ontology is depicted in Table 1.

Constraints:	**Number of VMs**	**Time Duration**	**Total Expenditure**	**Rectangle**
Relations:	=	=	Redundant	Redundant
	$\leq$	$\leq$	Optional	Required
	$\geq$	$\geq$	Required	Redundant

Table 1: Meaningful combinations of constraints for bids.

There are two types of bids in our system, namely future and spot bids. Future bids (or equivalently requests) are the bids for which the start and end times are fixed. For instance, a request could be: "User X bids for 5 VMs of type A to be used for 5 hrs, starting at time 13:00, with price 0.5 €/min/unit". As opposed to requests, spot bids demand to utilize resources as soon as they are available. Spot bids are distinguished from requests by setting the start time to a specific value (e.g. 0) and by the fact that

the start time and the end time are continuously moving as time passes. This is performed as long as the bid has not been matched (and up to the maximum time allowed by the expiry of the bid). For instance, a spot bid is: “User X bids for 5 VMs of type A to be used for 5 hrs, starting at time 0, with bid price 0.5 €/min/unit, and time limit 20:00”. In this example, the bid could be executed with a start time of 20:00 the latest.

An ask in our system describes the resources offered, which are specified by the following five parameters:

(1) The type of resources (i.e. VM) offered,
(2) The quantity of resources (i.e. number of VMs) offered,
(3) The start time and the end time of the interval when the resources are available,
(4) The price, expressed in €/min/unit, and
(5) The time limit for which the ask is valid (the expiration time of the offer). That means, the ask will be removed from the system after this time limit.

Similar to bids, there are also two types of asks, namely future and spot asks. Future asks (or equivalently offers) are those for which the start time and the end time are fixed instants in the future. For offers, the end time equals the start time plus the duration, while the time limit also has the same value by default. For instance, an offer looks like: “Provider Y offers for leasing 2 VMs of type A to be used for 8 hrs, starting at time 15:00, with a price of 0.2 €/min/unit”. On the contrary, spot asks offer resources that can be utilized as soon as there is demand for them. Such asks are distinguished from offers by setting the start time to a specific value (e.g. 0). Their main difference to offers is the fact that the start time and end time of asks are continuously moving as time passes. This continues as long as the ask is not matched (up to the maximum time allowed by the expiry of the ask). Therefore, spot asks are more flexible than offers. They offer service of a certain duration over a larger time interval. For instance, a spot ask is: “Provider Y offers 2 VMs of type A to be used for 3 hrs, starting at time 0, with an ask price of 0.2 €/min/unit, and a time limit of 19:00”.

3.4.3. Matching Algorithm

The matching algorithm defines how a bid (respectively request) in the spot market (respectively futures market) is matched by a set of asks (respectively offers). For simplicity reasons, it suffices to adopt a matching algorithm that passes the queues once. In our presentation below, we focus on the spot market. Indeed, the matching algorithm in the futures market is much simpler than that of the spot queue, since the time span of all requests and offers is fully specified, i.e. their start time and end time are decided upon their submission and cannot be changed subsequently, as opposed to spot bids/asks.

Trading in the spot market is performed by means of a continuous double auction mechanism. This is an extension of the standard spot market mechanism. Similarly to the standard mechanism, the spot bids and asks submitted by traders are placed in the *bid queue* and the *ask queue* respectively. Each queue is ordered according to the price and time of submission, with the *bid* queue being sorted in *decreasing* order of price, and the *ask* queue being sorted in *increasing* order of price. If two or more orders at the same price appear in a spot queue, then they are entered by time with

older orders appearing ahead of newer orders. An order remains in the queue until it is removed by the system due to order expiration, removal by the user, or if a matching had occurred.

Moreover, the matching algorithm takes into account that spot asks may start providing resources at some later time than now, due to the flexibility associated with the provision of resources. Note that we refrain from adopting a combinatorial approach due to the high computational complexity.

The matching algorithm for the spot offers is as follows: It initially computes the candidate matches to an ask by means of creating a matrix. Each column of the matrix corresponds to a time slot (i.e. the time interval in which service can be provided). Each row corresponds to a provider that can offer service now, with the cheapest being on the top row. A cell of the matrix is marked if the provider can offer computing resources during this specific time slot.

Then, the algorithm attempts to perform a probabilistic matching. In particular, the algorithm starts with the cheapest ask and randomly fills some time slots of bids, so that the provider's resource availability becomes zero. This means the cheapest ask is fully utilized. It then proceeds with the next cheapest ask and does the same. Note, after the second step, there might be slots allocated to two candidate providers. For these slots, each provider is assigned a probability of moving from this slot. A provider is moved to an empty slot according to a transition probability, which is larger for providers of this slot if they could serve a target slot where the number of providers that could serve the target slot is small.

The algorithm terminates when all the slots are assigned to some provider and thus a match is found. In case there are slots where there is no provider serving it, while there are not any slots with more than one provider, the algorithm has failed to compute a match. Due to the fact that we use a probabilistic matching algorithm, the algorithm can be repeated for a maximum pre-specified number of times until it terminates. If it fails, then it attempts to compute a match for the next time slot, i.e. for the time interval [Now + 1 slot, Now + 1 slot + service duration]. This is repeated until a match is indeed found or the algorithm fails for all time slots (i.e. entire duration) for which the bid is valid.

3.4.4. Routing of Bids and Asks

It is the responsibility of the matching module to be invoked periodically, prior to the beginning of the next slot, in order to compute matches and remove expired bids and asks from the bid/ask queue of the spot market and requests and offers from the futures directory of the futures market. The results of the matching procedure are subsequently passed to the scheduler, the reservation system of the provider, and the accounting system of the market place.

4 Conclusion and Discussion

In this paper, we discussed the rationale for a Grid market for leasing computing resources as well as the relevant key requirements. The GridEcon market place has one major advantage over existing utility computing services (e.g. Amazon's EC2

service, Sun, HP, and IBM). It allows companies not only to access computing resources, but also to sell spare computing resources. However, in the GridEcon market place, not all providers need also to be consumers and vice versa. Furthermore, the low market power of the participant of the Grid market ensures that the price, though flexible, remains highly competitive. Therefore, not considering market lock-in (i.e. high switching cost), network externalities, or anti-competitive behavior of market leaders, a group of many small computing resource providers (i.e. any company with spare computing resources) could compete with IBM, Google, and Amazon.

The market place for computing resources has certain similarities to the electricity market place. Indeed, since the market price is directly related to demand and supply, it will provide incentives for companies to adapt their usage strategies (e.g. buy more and own less computing power; compute during the night only). Moreover, companies will adjust their in-house computing usage to the competitive market price. Since the market place allows reselling resources that have been purchased, a company can buy resources on the market for a longer time period and resell those resources that are not needed at a shorter time scale.

Finally, the Grid market opens opportunities for a wide range of services (such as insurances, and capacity planning). Those value-added services on top of the market place will provide functionality that addresses certain needs of users. Such additional services may also be developed in some of the aforementioned existing utility computing services. However, it is likely that these services will be developed in such a way that it ultimately increases the profit of the provider, which is of course detrimental to buyers of the computing resources.

Acknowledgement. This work has been funded by the European Commission within the context of the FP6 Project GridEcon, Grid Economics and Business Models, (FP6-2005-IST5-033634).

5 References

[1] Amazon, Amazon Elastic Compute Cloud (Amazon EC2), http://www.amazon.com/gp/browse.html?node=201590011, June 2008.
[2] HP, Utility Computing Services, http://h20338.www2.hp.com/enterprise/cache/308072-0-0-0-121.html, June 2008.
[3] IBM, IBM E-Business On-Demand, http://www-128.ibm.com/developerworks/ibm/library/i-ebodov/index.html , June 2008.
[4] Sun Grid, http://www.sun.com/service/sungrid/index.jsp, June 2008
[5] Google, App. Engine, http://code.google.com/appengine/, June 2008
[6] NASDAQ Stock Market, http://www.nasdaq.com, June 2008
[7] International Monetary Fund, http://www.imf.org/external/np/pp/eng/2005/092105o.htm, May 2008
[8] Joskow P. L.: Restructuring, Competition and Regulatory Reform in the U.S. Electricity Sector. In: Journal of Economic Perspectives, vol. 11, no. 3, pp. 119-138, 1997.

[9] Dekrajangpetch S., Sheble G.B.: Structures and Formulations for Electric Power Auctions. In: Electric Power Systems Research, vol. 54, pp.159-167, Elsevier, 2000.
[10] GridEcon, http://www.gridecon.eu, June 2008.
[11] Altmann, J., Courboubetis, C., Darlington, J., Cohen, J.: GridEcon – The Economic-Enhanced Next-Generation Internet. In: GECON 2007, LNCS vol. 4685. Springer, Heidelberg, 2007.
[12] Techcrunch, http://www.techcrunch.com/2008/04/21/who-are-the-biggest-users-of-amazon-web-services-its-not-startups/ , May 2008.
[13] Altmann, J., Ion, M., Mohammed, A. A. B.: Taxonomy of Grid business models. GECON 2007, LNCS 4685, Springer, Heidelberg, 2007.
[14] XenSource, Inc., http://xen.org/, 2008.
[15] VMware, Inc, http://www.vmware.com/, 2008.
[16] PJM, http://www.pjm.com/contributions/pjm-manuals/pdf/m04.pdf, June 2008.
[17] European Energy Exchange, http://www.eex.com/en/document/32353, June 2008.

Taxonomy of Grid Business Models

Jörn Altmann[1,2], Mihaela Ion[1], Ashraf Adel Bany Mohammed[2]

[1]Intl. University, Bruchsal, Germany
[2]Seoul National University, Seoul, South-Korea
jorn.altmann@acm.org; mihaela.ion@i-u.de; ashraf@tepp.snu.ac.kr

Abstract. Grid Computing, initially intended to provide access to computational resources for high-performance computing applications, broadened its focus by addressing computational needs of enterprises. It became concerned with coordinating the on-demand, usage-based allocation of resources in dynamic, multi-institutional virtual organizations, and eventually creating new business models based on this technology. This trend in Grid computing holds a lot of potential in many industries with respect to saving costs, improving efficiency, creating new services and products, increasing product quality, as well as improving collaboration between companies. This will change the way business is done and it will change our classical view of the value chains, its stakeholders, and their roles. However, in order to encourage more companies to adopt Grid computing, value chains have to be explained and business models have to be understood. This paper makes a first move in this direction. It analyses existing business models. Based on the result of the analysis, it formally defines a taxonomy of existing and future roles that a stakeholder can take on within the value chains of the Grid and gives examples of those roles. Finally, this paper applies the taxonomy to two reference business models: utility computing and software-as-a-service.

Keywords: Grid Computing, Grid Economics, Business Models, Functional Roles, Taxonomy, Utility Computing, and Software-as-a-Service (SaaS).

1 Introduction

Grid Computing started out of the necessity to solve computational-intensive scientific problems that needed more resources than available at a single high-performance computing center (HPCC). Using Grid technology, storage capacity and processing power at several HPCC could be combined on-demand. As a next evolutionary step, the research in Grid computing broadened and became concerned with coordinating the allocation of resource in virtual organizations. This technology is based on virtualization of resources (i.e. processing power, storage capacity, bandwidth, and data). It makes distributed resources available to the user as a single unified system. In general, organizations using Grid technology can optimize the use of their departmental resources by sharing them across departments, run computational-intensive applications on their Enterprise Grid, and even enable collaboration with other organizations [8].

Although the Grid could potentially offer a more efficient way of developing products and creating new business opportunities, the use of the Grid is quite limited at present. Grid Computing is mostly used as a mean for simplifying resource management. Because of that, only a small number of companies are deploying Grid-related technologies and none of the small companies (SMEs) consider using the Grid at the moment [9].

To make the Grid being adopted, companies need to understand the benefits they could gain from using the Grid. The support that they need is a clear analysis of the value chains and the cost models of the Grid. Both will help them understanding the real cost cuts (i.e. the amount of money and time that could be saved using the Grid).

In addition to this, the analysis of incentives and concrete business models is needed. A Grid business model defines a framework for creating new value chains. The analysis of Grid business models will show providers and consumers how to trade resources and software services on the Grid. This opens up opportunities for creating new businesses and revenue streams.

Within this paper, we present the results of our analysis of a set of Grid business models. In particular, we give an overview of existing business models and projects investigating Grid business models in chapter 2. Chapter 3 introduces the taxonomy. It classifies and defines the roles that stakeholders could take on within the Grid. The roles describe atomic functions that could be the basis for new value chains on the Grid. Two examples for using the taxonomy are given in chapter 4. It demonstrates the usefulness of the taxonomy and explains in more detail two abstract business models (i.e. utility computing and software-as-a-service). Finally, the conclusion is given in chapter 5.

2 Classifications of Grid Business Models

2.1 Existing Business Models

The existing Grid business models can be classified according to their origin in research or in commerce. The business models of the research category have mainly been developed by universities and research centers. These business models are based on an open Grid architecture that would allow several providers and consumers to be interconnected and to trade services. The business models of the commerce category have been developed and deployed by a single company with the purpose of selling its own products. These business models usually do not involve several providers.

Research Business Models. The following research projects on Grid business models were examined: GridASP [5], GRASP [6], GRACE [7], and BIG [1]. These projects promote open value chains for trading services on the Grid.

GridASP and GRASP rely on the concept of an application service provider (ASP) for delivering and composing services on the Grid. GridASP offers a scalable and service-oriented architecture (SOA), which offers a lot of potential for creating new

products and business models. The business model involves only four roles: the consumer, the service provider (which functions as a portal for the consumer, basically aggregating applications and resources), the application provider (which offers the use of software applications), and the resource provider (which owns hardware resources). From a technical perspective, GridASP addresses all relevant aspects: user management, data and job management, workflow handling, resource brokering, semi-automated application deployment, and security [2][3]. GridASP lacks a better integration of economic functions such as SLA management, negotiation of services, accounting, capacity planning, and pricing.

GRASP also offers a scalable and service-oriented architecture focusing on Web Services and OGSA standards, which offer a good support for service integration. The main focus of GRASP is to allow innovative business models and to integrate economic functions such as accounting, billing, and SLA management into the architecture. As opposed to GridASP, the GRASP architecture offers better support for collaboration between organizations and ASPs by allowing to create virtual organizations and a federation of ASPs.

Compared to GridASP and GRASP, GRACE is very economic-oriented and less focused on architectural issues, aiming to develop a generic framework or infrastructure for a computational Grid economy. The framework provides brokering, service discovery, and trading through an innovative API for negotiating prices and services on the Grid. GRACE relies on existing Grid middleware such as Globus [10] and Legion [11]. GRACE defines only two main roles in exchanging services on the Grid: the consumer, represented by the broker, and the seller or resource owner.

The BIG project addresses the problem of Grid business models from a more general and theoretical perspective [1][4]. It classifies the current Grid projects in four levels based on the support for economical functions and business models. It also focuses on requirements for innovative business models on the Grid and identifies transparency, QoS, brokerage, SLA, and dynamic trust management in virtual organizations to be the most important requirements. BIG supports a large set of innovative applications such as dynamic collaborations, workflows, applications on demand, dynamic resource-management, and resources on demand.

Commercial Business Models. The following commercial models were analyzed: Sun Grid Compute Utility [12], Amazon EC2 [13], the Virtual Private Grid (VPG) [15], and WebEx Connect Application Grid [14]. Both, Sun Utility Grid and Amazon EC2, provide on-demand computing resources while VPG and WebEx provides on-demand applications.

Sun Utility Grid allows the user to create jobs and submit an application, but does not give the user means to control or monitor the execution. The logs and results, together with error reports are provided once the execution is completed. Amazon, on the other hand, allows the user to create virtual machines, which makes the job execution fully transparent. The user has access to virtual machines with 1.7Ghz Xeon CPU, 1.75GB of RAM, 160GB of local disk, and 250Mb/s of network bandwidth. Users can initiate, run, and monitor applications on each virtual machine. Additionally, Amazon provides storage through Amazon S3. For both models, the user is only charged for the consumed resources. Sun Grid Compute Utility charges $1/CPU-hr. Amazon charges $0.10 per instance-hour consumed (or part of an hour

consumed), $0.20 per GB of IP data transferred into and out of Amazon, and $0.15 per GB-Month of Amazon S3 storage (as of February 2007).

The VPG has been designed by the British Telecom (BT) together with industry partners [15]. The VPG will allow BT to provide its customers with services such as music or video on-demand. The services and resources are developed and are owned by BT's partners. The VPG is based on virtualization of resources and a SOA architecture. This allows combining resources of application and resource providers. However, VPG does not support complex applications such as workflows and has no support for composition of services. Economic functions also lack. Another shortcoming of this model is that BT has full control over the network, and manages the resources and applications, acting as the only reseller. The VPG does not allow other network providers and resource brokers to coexist on the same Grid.

WebEx provides online meeting, web conferencing, video conferencing, and teleconferencing for enterprises. Their software-as-a-service (SaaS) implementation allows innovative and complex, on-demand composition of services and workflows. Moreover, WebEx enables developers to attach new applications to the Grid and sell their products to customers through the WebEx platform. The applications are delivered through their WebEx MediaTone Network, a private global network and platform. MediaTone Network includes connected data centers and servers distributed around the world. However, WebEx only focus on a specific kind of application, namely Web meeting software [14].

2.2 State-of-the-Art in Grid Business Model Classifications

The identification of stakeholders and their roles in Grid business models has been addressed by many scholars [16][17][18][19][20]. Our work harmonizes these classifications and builds our taxonomy of Grid business models (see section 3) on top of it.

Some classifications of classical Internet service providers can be found in [17] and [19]. Their classifications of roles are based on business transactions and are organized in layers. The Software-as-a-Service business was classified in [16]. Another classification of Internet businesses can be found in [18], where the authors illustrate a five-tier classification. Standard bodies, consortia, academic groups of interest, and governments are at the lowest layer, setting the rules for collaboration. This layer is followed by the layer of large technology vendors, niche vendors (who integrate), and application vendors. Within the third layer (influenced by media and information sources), there are consultants, resource service providers, and resellers, which provide customized services to the next layer consisting of business users and retail service providers. This layer provides the provisioning to the last, the fifth layer, consisting of end-users.

In more detail, Grid has been analyzed in [20]. Their model of Grid businesses focuses on the structure of Grid-aware markets. Its layers are divided into two main groups: the Grid market participants and the technology enablers. The Grid market participants consist of three tiers: the service tier (consisting of service providers, content providers, consolidators), the platform tier (consisting of Grid infrastructure providers such as Grid operators and resource providers), and the consumer tier

(consisting of partner Grids, virtual organizations (VOs), enterprise Grids, department Grids, and end-users). The technology enablers are also organized into three tiers with the same name as the tiers of the Grid market participants. However, the service tier consists of application providers, the platform tier of middleware vendors, and the consumer tier of consultants and integrators.

3 Roles and Stakeholders

This chapter presents a classification of roles that stakeholders on the Grid could take on. The definition of stakeholders and roles that we follow is: A Grid *stakeholder* is an entity that takes on one or several *roles* in a business model for selling Grid services. A *grid service* is defined as any service that can be provided on the Grid.

Figure 1 shows a classification of roles. There are five categories of roles that a stakeholder could assume on the Grid. Those categories are the roles of a Hardware Resource Service Provider, a Grid Middleware Service Provider, a Software Service Provider, Content Provider, and a Consumer.

I. **Hardware Resources Service Providers:** This is the lowest layer of the classification representing hardware providers. The hardware can belong to many different providers. In detail, this layer includes:
   - **A. Storage Resource Providers:** This role represents the stakeholders providing huge storage systems or a collection of physical or virtual storage resources (located on geographically distributed PCs). Examples are systems such as Amazon's Simple Storage System (S3) or Openomy.
   - **B. Computing Resources Providers:** They provide computing resources such as Amazon (Elastic Compute Cloud (EC2)) and Supercomputing centers.
   - **C. Network Services Providers:** This layer represents the network providers including ISPs and their multi-tier classification as explained in more details in [17].
   - **D. Devices Service Providers:** They provide the access devices such as sensors and microscopes.

II. **Grid Middleware Service Providers:** The stakeholders in this layer provide the services build upon the above-mentioned physical layer. This role is comprised of two distinct roles:
   - **A. Basic Grid Middleware Service Providers**: This role represents the collection of stakeholders providing basic Grid functionality. This layer includes the following four main roles:
     a. *Grid Resource Management Service Providers*: This role comprises the stakeholders that provide the following management functionalities:
        1. Resource discovery services.
        2. Resource allocation management services and virtualization.
        3. Resource connectivity services.
        4. Metering and monitoring services.
        5. Job scheduling services.
     b. *Security Services Providers*: This role comprises the stakeholders involved in providing security functionality.

c. *Fault Tolerance Service Providers:* This role consists of error detection, error recovery, job mapping, and check pointing services.
d. *Grid Billing Management Service Providers:* This role covers the entire billing stack for any kind of service (1. Accounting services. 2. Charging services. 3. Pricing systems services. 4. *Payment* management services.)

B. Composite Resources Service Providers: This layer represents the role played by stakeholders providing a value-added composite service, which includes services of the previous layers. The stakeholders in this layer take one or more of the following roles:

a. *Service Level Agreement Services (SLAs) Providers:* They provide services for the contract (SLA) management, negotiation, monitoring, and auditing.
b. *Grid Services Brokers:* Based on the specific task that they provide, brokers can be classified as:
    1. *Risk Brokers*: These brokers minimize the cost for consumers by finding not only the best deal from several offers based on user specified parameters but also based on the uncertainty of the availability of resources.
    2. *Trust Brokers*: They help users of the Grid to assess the uncertainty, which results from using resources of unknown providers.
    3. *Value Brokers:* They perform the task of managing jobs (even entire workflows related to Grid) on behalf of a consumer. This role can also be further sub-divided based on the kind of job (e.g. finding hardware resources and *composing resources).*
c. *Capacity Planners:* They assure in the long term that the balance between demand and supply is met. In order to maximize their utility, they calculate how many and when to buy / sell resources.
d. *Market-Place Providers:* Stakeholders in this role provide a market place for trading Grid services. These services can be hardware resources, basic grid middleware services, and composite resources.
e. *Grid Service Resellers*: They provide selling and retail services of Grid-oriented services.

III. Software Service Providers: A stakeholder in this role takes on one or more of the following sub-roles dealing with software:

A. Application Service Providers: These stakeholders provide (commercial or open-source) application services, either ready to use of-the-shelf packages or customized services. Applications (services) include all types of application such as multimedia, scientific, and business applications. Types of application service providers are:

a. *Software-as-a-Service (SaaS) Providers.* An example is the execution and maintenance of an Apache Web server.
b. *Software Repository Providers*. This provider maintains a repository of software and controls access to this software.
c. *Software Hosting Providers:* This type of stakeholder provides the software *environment* for applications to be executed.

B. Billing Management Service Providers: They are equivalent to the providers of II.A.d.

C. **Software Market-Place Providers:** Stakeholders in this role provide the market place for software services.
D. **Software Brokers:** Stakeholders in this role provide the brokering services for software services. They are similar to II.B.b.
E. **Software Resellers and Retailers:** They sell software services.
F. **Applications-to-Grid Wrappers:** They provide integration of applications into the Grid.
G. **Software Vendors:** This stakeholder develops software. They can be:
    a. *Open Source Software Developer:* The source code of this software is available under certain licensing conditions.
    b. *Commercial Software Companies:* This software is mainly proprietary. The use of the software is restricted.
    c. *Software Integrators:* These stakeholders develop software to interface different software components.

IV. **Content Providers:** This layer represents *roles* available on the information side of our model. Stakeholders in this layer can be divided into the following groups:
A. **Content Creators:** Any entity that creates content, regardless of the content type (e.g. photo, video, and text).
B. **Content Aggregators:** They aggregate, classify, and organize content for either business or individual use, using applications such as *mediawiki*, tags as in *clipmarks*, or bookmarking services as *blinklist*.
C. **Content Composers:** They re-build and modify the content. They add some value and then resell it. An example is the company *programmableweb.com*.
D. **Content Distributors:** This type includes content update disseminators. Examples are *RSS* services as in *Tailrank* and *Topix.net* as well as traditional content distribution channels such as classical Web media channels.
E. **Content Brokers:** Stakeholders in this role provide matching services between content providers and consumers.
F. **Content Resellers and Retailers:** They provide retail services for content (e.g. flicker).
G. **Content Market-Place Providers:** They provide the market place where content can be exchanged according to some economic rules (e.g. *Flicker*, *YouTupe*, and *Democracy 2.0*).

V. **Consumers:** This layer represents roles taken on by stakeholders who simply consume a service. Three major types of consumers can be distinguished:
A. **Business Users:** This entity is either a virtual or physical business entity; seeking value-added services. They can be classified into:
    a. *Core-Business Consumers*: They seek core business process applications. Examples of those stakeholders include: pharmaceutical companies (run applications for drug discovery), financial institutions (run complex applications to make accurate financial estimations).
    b. *General-Business Consumers*: They seek general business process services (i.e. outsourcing of storage, CPU, content development). Examples of stakeholders can be any government agency and business.
B. **End-Users:** This entity is a single person or group, which consumes the services without the intention of modifying or reproducing the application.
C. **Universities:** This entity has non-monetary objectives when using resources.

In addition to this classification, we need to mention the emergence of the following three roles. These roles provide additional services for stakeholders, spanning over the above-described layers (Figure 1):

- *Grid Consultants:* This role provides consultation service (such as economic analysis, technical analysis, education, and training) for Grid adopters. Examples of stakeholders are software integrators and strategic consultants.
- *Grid Standardization Bodies*: This role provides standardization services, which could be taken on by stakeholders such as academia, governments, and consortia.
- *Regulators*: This role will guide the development of the Grid through policies.

To complement our classification of roles on the Grid, we need to mention the following four facts about stakeholders that take on those roles:

- Any provider can offer integrated services through *horizontal service integration* (i.e. integration roles of the same layer) and / or *vertical service integration* (i.e. integration of roles of different layers). The integration of Grid technology, Web services, and Web2.0 enables this. It will give each stakeholder the potential to change his role by adding / deleting more roles to his stake and build new business models. It will allow all stakeholders to adapt quickly to new market conditions.
- Service providers can become consumers of services and vice versa.
- Not all of the providers mentioned have to be present in the future market. Each of the providers can serve a different niche market.
- Even though the layered structure is the ideal case, the stakeholder relationships between the layers can follow different paths. The path that a consumer takes to use a service does not have to go through all the layers. The consumer can directly choose the most appropriate service from any layer. However, the services of different layers have to be ordered according to the layered structure.

Figure 1 summarizes the roles of the stakeholders and depicts the relationships between them. The arrows represent the direction of service delivery. They are used to indicate the service that a stakeholder delivers to another stakeholder in a different role. The following chapter analyzes two business models using this taxonomy.

4 Role Analysis of Two Reference Business Models

We discuss two abstract business models with respect to the service functionality that they require for their implementation. The first reference business model is “Economically Efficient Utility Computing”. In this case, the user owns the software that will be executed on the Grid. The second reference business model is “Software-as-a-Service”, which explains the value chain of software on demand. In this case, the user rents the software and has the option to specify the hardware resources on which the software should run.

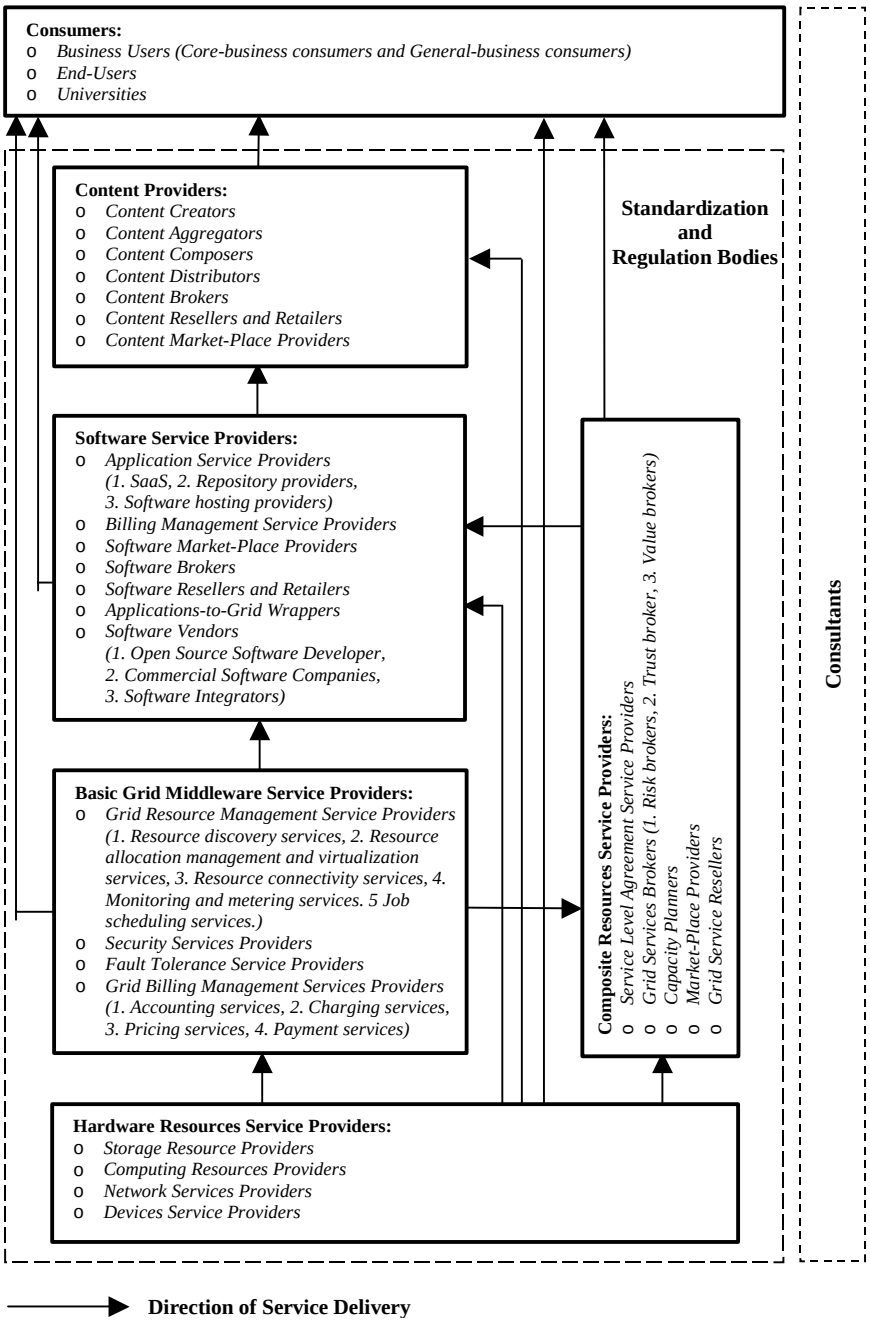


Fig.1. Roles of stakeholders and their relationships

4.1 Reference Business Models: Economically Efficient Utility Computing

There are a number of advantages of utility computing. First, different Grid systems that provide utility computing can perform load balancing amongst each other, thus ensuring that the capacity of the Grid is used to the highest extent possible. Second, utility computing implies that computational power is always present. The Grid is inherently fault-tolerant. (It is unlikely that all single Grid systems will fail on the Grid at the same time.) Furthermore, the operation of the Grid is transparent, meaning that the user is not aware of the fact that the application is running on a geographically distributed system. All these advantages do not only enable the execution of computationally intensive, scientific applications but also allow commercial customers to use the power of such a Grid to solve their problems quickly and efficiently.

However, there are many different kinds of users (e.g. SMEs, large enterprises, the general public, and academia), distinguishing themselves in the amount of budget, urgency of their application, and quality of service expectations. For example, industry users, which try to achieve a competitive advantage, require the termination of the execution of their application within a specific period of time. Because of that they are willing to pay a higher price than other users. In order to get their job executed on time, i.e. get priority over other users, the Grid could charge them a premium fee, which is specified in an agreement between the user and the Grid system. This agreement, which is called a Service Level Agreement (SLA), states what the user wants and what the provider promises to supply in a legally binding way. An SLA also describes a measurable performance standard and a penalty fee if it is not delivered.

This paradigm of utility computing supports the execution of any application (i.e. high-performance computing workflows, parallel applications, or simple sequential application, such as Web servers). Any end-user can get access to any kind of computational resource, ranging from supercomputers, which provide large computing power, to a single PC. Figure 2 shows the interaction between the consumer and the Grid. In this case, the consumer owns a software application and executes it on the Grid. The Grid offers the mechanisms for deploying and executing the application (e.g. automatic deployment, execution monitoring, and hardware resource discovery). Figure 2 also illustrates the business processes (or service interactions) for purchasing hardware resources on the Grid and for executing the application. Implementing such a business model requires at least the following basic roles, which belong to three layers:

- **Hardware Resource Service Providers:** For the utility computing business model, server, storage, and network resources are considered.
- **Grid Resource Service Providers:** This intermediate layer between the Consumer and Hardware Resource Service Providers offers only services needed to execute an application:
    - **Basic Grid Middleware Service Providers:** The basic Grid middleware must provide at least security, and accounting.
        - **Billing Stack Service Provider:** Interacts with the Resource Broker to charge the Consumer for consumed resources. It has a Pricing component for storing current and past prices and components for

accounting, charging, and billing. When the execution completes, it bills the consumer.

- **Grid Resource Management Service Providers:** It provides virtualization, metering, job deployment, and resource discovery.

- **Composite Resource Service Providers:** In this example of utility computing business models, only one type of provider is needed:
  - **Resource Broker:** It selects the best-fitting resources from different Hardware Resource Service Providers upon user request and initiates the job deployment and monitors the job execution.

- **Consumers:** The consumer runs its application on the Grid, using the services of the Grid Resource Service Provider. The consumer can be of any type.

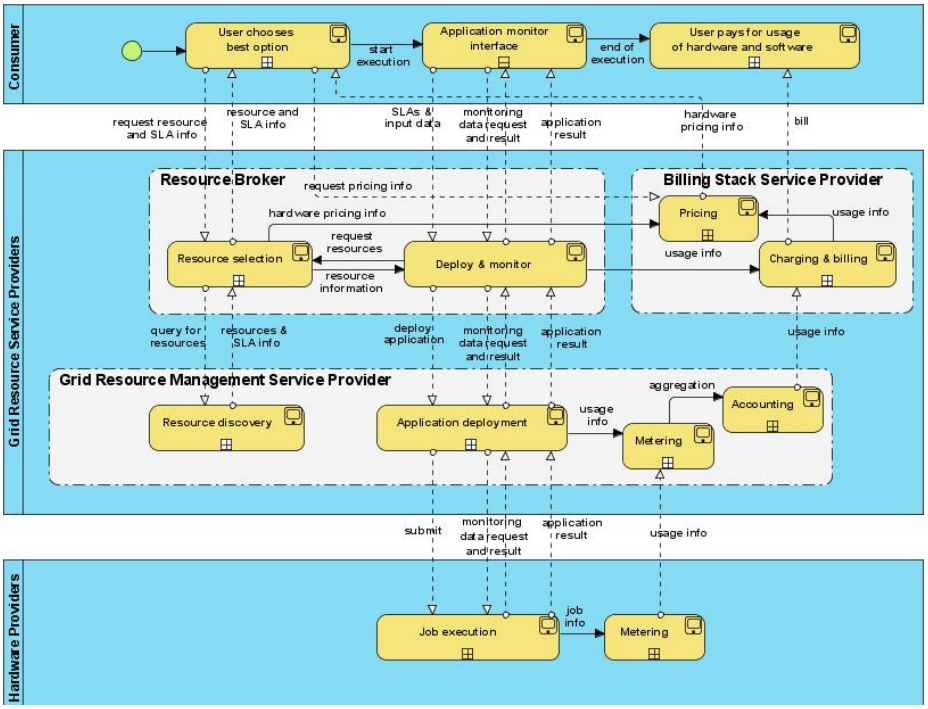


Fig. 2. Interaction of roles in the utility computing business model

A typical scenario of this business model can be found in high-performance computing. Applications in this scenario require huge amount of hardware resources. Applications range from scientific (simulations such as weather and climate modeling, weather prediction), digital media (animation, special effects, rendering), life sciences and health care (drug discovery, structure-based design, molecular dynamics, medical imaging), financial services (Monte Carlo simulations, risk analysis) to manufacturing.

4.2 Reference Business Model: Software-as-a-Service (SaaS)

For software owners, this model has a number of advantages. First, they no longer need to be concerned with license agreement violations, which are common when software is sold directly to the customer. Second, since consumers pay only for their usage, even consumers, who cannot afford buying very expensive software, can now purchase units of software usage. These additional customers will increase the amount of income for the software owner and, thus, contribute to higher profits from software development. However, the following third reason is the most important one for software vendors. Since the software vendor can use the Grid to run its software, they do not need to own the hardware resources themselves. Instead, they can reserve a set of hardware resources for a long time period on the Grid and, should more hardware resources be needed, buy additional resources on demand. This degree of flexibility cannot be achieved without the Grid (i.e. without sharing of resources). It makes the Grid an attractive alternative to buying and maintaining hardware resources.

The SaaS business model also has advantages for customers. First, customers do not have to buy additional, sometimes even highly specialized hardware resources to run purchased software anymore. This does not necessarily reduce the cost of hardware resources, but reduces the cost of ownership of software and hardware for customers. Therefore, the pay-per-use model opens the opportunity to use the most powerful software, which would otherwise be too expensive to buy. For customers, such as SMEs, SaaS levels the playing field when it comes to competing with large companies. This aspect is especially important for SMEs in fields like metallurgy, which require highly complex computations, and for SMEs, which specialize in customized products in niche markets.

This reference business model, Software-as-a-Service, involves the purchase of software and hardware resources. Figure 3 shows the business process of running a SaaS business model on the Grid. The basic services that a system has to provide to offer software-as-a-service belong to four layers:

- **Hardware Resource Service Providers:** This layer consists of servers, storage, and network capacity that are required to run the SaaS software.
- **Grid Resource Service Providers:** The services offered within this layer are identical to the services offered by the Grid Resource Service Providers of the utility computing business model.
- **Software Service Providers:** For the SaaS business model, this layer represents service providers who maintain software, which they do not necessarily own and execute on Grid resource hardware. The providers considered here are:
    - **Software Discovery Provider:** Software vendors have registered their applications in a Software Registry. Information about available software applications and their SLA can be retrieved from the registry.
    - **Software Broker:** It uses a Software Discovery Service to retrieve information about similar applications that match the consumer's preferences. It then offers a selection to the consumer.
    - **Application Service Provider:** In the SaaS business model case, it is the environment, in which the SaaS software is executed on the Grid.
    - **Billing Management Service Provider:** Interacts with the Application Service Provider and with the Billing Stack Service Provider of the Grid

Resource Service Provider layer to provide services for billing the consumer for the consumed hardware resources and the application usage. When the execution completes, it bills the consumer.

- **Consumers:** This entity is the consumer, who uses the software. It buys access to the software on a usage-basis, using the services provided by the Software Service Providers. It can be a SME or an individual.

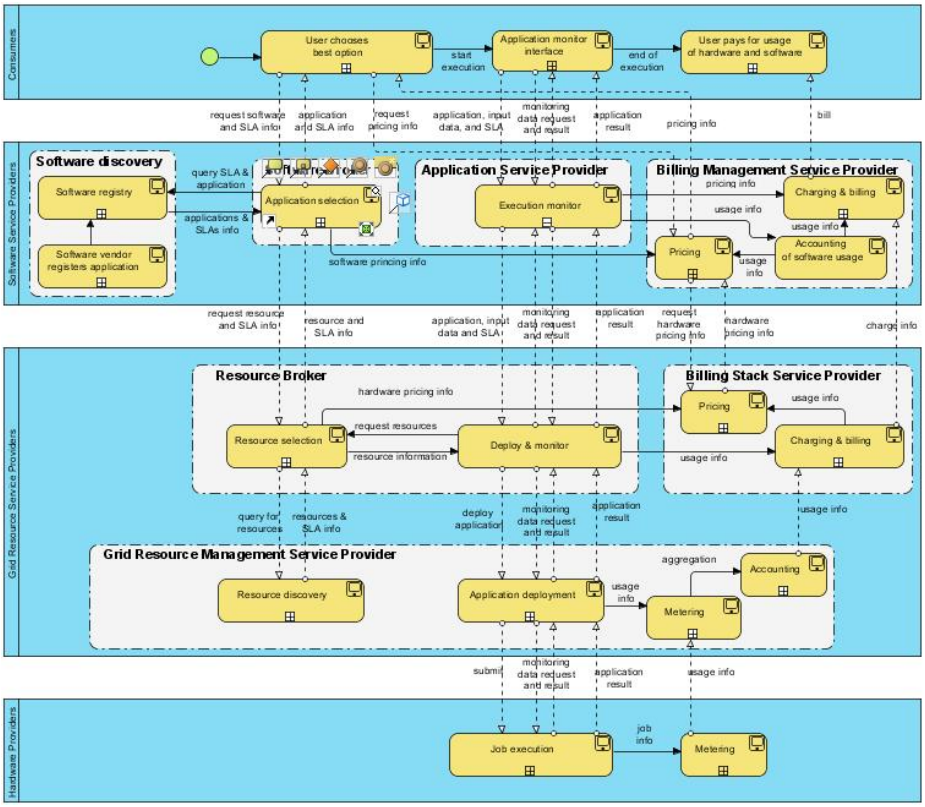


Fig. 3. Interaction of roles in the software-as-a-service business model

The SaaS business model allows SMEs to gain access to expensive commercial software that they could not afford to purchase licenses for or because they do not have the expertise to develop the application inside the company. Another interesting scenario is the one of starting a business with little investment by using the software services (or a composition of them) provided on the Grid and adding own expertise. For example, an interior decorator could use rendering and visualization software to provide advice on house decoration.

5 Conclusions and Future Work

This paper presented a survey of current Grid business models. It then identified the roles, which can be assumed by different stakeholders on the Grid, and classified those roles into five groups, defining the taxonomy of Grid business models. The groups are: Hardware Resource Service Providers, which own storage, network capacity, devices, and server capacity; Grid Resource Service Providers, which provide the Grid middleware and composite resource services; Software Service Providers, providing the software and the environment for managing and executing software on the Grid; Content Providers, which create, aggregate, and compose information; and, finally, Consumers, which are entities who consume services on the Grid. It is to note, that a stakeholder can assume multiple roles as well as roles within different groups.

In addition to this, we discussed two reference business models and applied the taxonomy to those two business models. These reference business models are "Economically Efficient Utility Computing" and "Software-as-a-Service". In the first case, the user owns the software that will be executed on the Grid. In the second reference business model, the user pays for the usage of the software and the Grid hardware resources. The business processes, which are represented through interactions between roles, give a general guideline for what is needed to implement those business models on the Grid with respect to the service functionality required.

After having made the first step with this analysis of existing business models and roles, the incentives for deploying and using the Grid need to be investigated. In particular, the impact of different pricing schemes for service-oriented computing has to be investigated. This will provide more insight into value chains of Grid businesses.

References

1. T. Weishäupl, F. Donno, E. Schikuta, H. Stockinger, and H. Wanek, "Business In the Grid: The BIG Project," GECON2005, the 2nd International Workshop on Grid Economics and Business Models, Seoul 2005.
2. S.Itoh, H. Ogawa, T. Sonoda, and S. Sekiguchi, "GridASP - A framework for a new utility business," In proceedings of GECON2005, the 2nd International Workshop on Grid Economics and Business Models, Seoul 2005.
3. S.Itoh, H. Ogawa, T. Sonoda, and S. Sekiguchi, "GridASP: An ASP Framework for Grid Utility Computing," http://www.teragrid.org/programs/sci_gateways/docs/gridasp-ogawa.pdf, 2005.
4. BIG, Faculty of Computer Science, University of Viena, Web site, http://www.cs.univie.ac.at/project.php?pid=79, 2005.
5. GridASP Web site, http://www.gridasp.org/en/, 2007.
6. GRASP Web site, http://eu-grasp.net, 2007.
7. GRACE Web site, http://www.gridbus.org/ecogrid/, 2007.
8. IBM Grid Computing, Web site, http://www-1.ibm.com/grid/about_grid/what_is.shtml, 2006.

9. E. Schikuta, F. Donno, H. Stockinger, E. Vinek, H. Wanek, T. Weishäupl, and C. Witzany, “Business In the Grid: Project Results,” http://www.pri.univie.ac.at/Publications/2005/Schikuta_austriangrid_bigresults.pdf, 2005.
10. Globus Web site, http://www.globus.org/, 2007.
11. Legion, Web site, http://legion.virginia.edu/index.html, 2007.
12. Sun Grid, Web site, http://www.sun.com/service/sungrid/, 2007.
13. Amazon Web Services, http://www.amazon.com/b/ref=sc_fe_c_1_3435361_1/104-3351913-5483951?ie=UTF8&node=201590011&no=3435361&me=A36L942TSJ2AJA, 2007.
14. WebEx Connect, “First SaaS Platform to Deliver Mashup Business Applications for Knowledge Workers,” http://www.webex.com/pr/pr428.html, 2007.
15. F. Falcon, “GRID – A Telco perspective: The BT Grid Strategy,” GECON2005, the 2nd International Workshop on Grid Economics and Business Models, Seoul, 2005.
16. E. Traudt and A. Konary, “Software-as-a-Service Taxonomy and Research Guide,” http://www.idc.com/getdoc.jsp?containerid=33453&pagetype=printfriendly#33453-S-0001, 2005.
17. J. Altmann, “A Reference Model of Internet Service Provider Businesses,” ICTEC2000, 3rd International Conference on Telecommunication and Electronic Commerce, Dallas, Texas, USA, November 2000.
18. M. Bryson, “What To Do When Stakeholders Matter: A Guide to Stakeholder Identification and Analysis Techniques,” London School of Economics and Political Science, February 2003.
19. J. Altmann and S. Routzounis, “Economic Modeling for Grid Services,” *e-Challenges 2006*, Barcelona, October 2006.
20. P. Plaszak and R. Wellner, Jr., “Grid Computing: The Savvy Manager’s Guide,” Elsevier, 2006.

Pricing Resources on Demand

Costas Courcoubetis*, Sergios Soursos* and Richard Weber†
*Athens University of Economics and Business
Email: {courcou, sns}@aueb.gr
†University of Cambridge
Email: rrw1@statslab.cam.ac.uk

***Abstract*— Traditional contracts for network and computing resources are of "static" type where the customer is buying the right to use for a given price a fixed amount of resources for a long period of time. Typical examples are the case of contracting bandwidth in access networks and VPNs and the case of computing infrastructure that a customer leases (or buys) for fulfilling its IT needs. Current technology in access networks and Grid computing allows suppliers to offer more flexible contracts to their customers allowing them to choose dynamically the amount of resources they are allowed to use at a given time. This flexibility may benefit the customers with bursty demand since it allows them to obtain resources only when they need them and pay only when they use them. We define contracts where time is discrete and a customer is allowed to buy a fixed amount of resources ahead of time for a price a, the "static" part of the contract, and complement this at each new time period by purchasing an extra amount at price b, the "dynamic" part of the contract. We investigate the properties of such contracts and compare them with contracts of purely static or dynamic type. Our results suggest that in general suppliers and customers are both better off when using such mixed contracts, and that purely dynamic contracts may not always be preferable compared to purely static ones. We also show that under price competition of suppliers using static contracts against suppliers using dynamic contracts, at the equilibrium both suppliers may secure some profit by segmenting the market.**

I. Introduction

Traditional bandwidth contracts have been for fixed bandwidth pipes and are long term, typically a year. We are thinking of a network supplier that leases high capacity broadband lines to other large companies, such as banks or Internet service suppliers (ISPs). A bank might lease a 155Mbps line between two locations for one year. An AOL subscriber might sign a year's contract for the service known as 'AOL Broadband Gold' (2Mbps). Typically, such Internet access or VPN services do not offer any flexibility to the customer to alter dynamically the size of the pipe. Similarly, when acquiring computing resources such a servers and PCs, a company's IT department usually makes yearly leasing contracts for fixed amount of computing resources, or buys these resources which is the equivalent of a fixed size contract for an even longer period of time. The problem customers face in making such contracts is that their demand for resources is not constant but bursty and in many cases unpredictable. They hence face the risk of acquiring for a long time period resources that may not be effectively used, or acquiring fewer resources that will fail to meet high short term demand and peak load requirements.

Emerging technology in access networks and Grid computing allows suppliers to offer more flexible contracts to their customers providing them the ability to choose dynamically the amount of resources they are allowed to use at a given time. For instance, DSLForum's BroadbandSuite platform [1] and IPsphere's framework [2] propose solutions that will offer the ability to alter the size of the bandwidth pipe with which a customer is provided on a time scale that is much finer than yearly. They also provide accounting and billing for such dynamic services. Grid computing offers a direct analogy to bandwidth provisioning. Current Grid utility computing architectures such as SUN Grid Compute Utility [3] allow customers to buy computing cycles on demand while also using their existing computing facilities. This flexibility should benefit the customers with bursty demand since it allows them to obtain resources only when they need them and pay only when they use them. If priced correctly, these services may increase the revenue of the provider by obtaining a share of the added value to his customers.

In this paper, we investigate the properties of a new type of contract, the "mixed" contract, which is composed by combining static and dynamic contracts. Under a mixed contract a customer is allowed to buy a fixed amount of resources ahead of time for a price a, the "static" part of the contract, and complement this at each new time period by purchasing an extra amount at price b, the "dynamic" part of the contract.

Dynamic contracts or the dynamic part of a mixed contract should not be confused with *dynamic pricing approaches* which aim to control network traffic by means of prices. One such approach is *peak-load pricing* [4], [5] where prices can adjust to the fluctuations of demand and may reflect the investment made in order to serve the high-demand periods. Due to the fact that mixed contracts have two parts, they may be considered as *two-part tariffs*. In the literature, two-part tariffs are used by the providers to obtain a greater portion -if not all- of a customer's utility. The static part of such a tariff is a lump sum that the user must pay in order to have the right to use the resources [6] or a way for the provider to cover his fixed costs. In some cases, this lump sum may give the right to use a small amount of resources. But in our pricing scheme, no lump sum and no fixed amount of resources is introduced. Our two-part contract aims to make more flexible the customer in the way he expresses his bandwidth needs. Throughout the paper we consider that both prices (a and b) are fixed, i.e. they

This work is part of the project "Economic and Architectural Aspects for Services of Differentiated Quality in high-speed Broadband and Wireless Access Networks", co-funded by the "European Social Fund & National Resources - EPEAEK II - IRAKLITOS Fellowships for research of Athens University of Economics and Business". This work is also supported by the IST project GridEcon (IST-2006-033634) funded by the EU.

are not affected by the demand. On the other hand, these prices can be considered as usage-based prices since customers are charged based on the volume of bandwidth they consume.

An interesting question we analyze in this paper concerns the position of the suppliers of dynamic services. Will they be always better off by providing such services or they may cannibalize their lucrative static contracts? Will bandwidth consumption increase or decrease? What will be the result of price competition between dynamic and static service suppliers? Will one supplier get the whole market share by displacing the other? We show that many interesting facts may occur. For instance, in a monopolist situation, a customer may not always be better off when the supplier offers purely dynamic contracts. Also our models suggest that resource consumption may drop when dynamic contracts are used (which justifies the fears of the network operators for allowing dynamic contracts). Moreover, we show that offering mixed contracts (the "mixed" suppliers) is always beneficial to both the customers and the resource suppliers. Also, under price competition, the market will be segmented and both the dynamic and static suppliers may make profit at the equilibrium.

A simple example

Let us suppose, for example, that at the start of a year a customer can sign a contract for a fixed line at a cost of $\$a$ per 1 Mbps. Additionally, he may purchase at the start of each week, additional capacity for a cost of $\$b$ per 1 Mbps per year. Assuming $a < b$, there is the incentive to book ahead. This is easier for customers whose requirements are more predictable. Suppose that the supplier has no capacity constraint. Consider now three customers and their bandwidth requirements. For simplicity we assume that these requirements are inelastic.

Customer 1 needs 1 Mbps pipe throughout the year. The most he would be prepared to pay for this is \$100. Customer 2 needs a 1 Mbps pipe on half of the weeks of the year and a 2 Mbps pipe on the other half of the weeks of the year. These weeks are randomly distributed in the year, but are predictable one week ahead. The most he would be prepared to pay for this is \$190. Customer 3 needs a 1 Mbps pipe on $9/10$ths of the weeks of the year and a 3 Mbps pipe on the other $1/10$ths of the weeks of the year, which are again randomly distributed in the year and predictable one week ahead. The most he would be prepared to pay for this is \$ 136. Customers 2 and 3 will only purchase a contract if it can satisfy their requirements on all weeks of the year. The problem for the supplier is to choose a and b so as to solve the problem

$$\text{maximize} \sum_{i=1}^{3} y_i,$$

subject to $y_1 = a$ if $a \leq 100$, otherwise 0, $y_2 = \min\{2a, a + b/2\}$ if $\min\{2a, a + b/2\} \leq 190$, otherwise 0, and $y_3 = \min\{3a, a + 2b/10\}$ if $\min\{3a, a + 2b/10\} \leq 136$, otherwise 0.

If we allow only the year-long contracts, (so that effectively $b = \infty$), then by taking $a = 95$ the revenue is maximized to $95 + 190 = \$285$. Only the first two customers purchase contracts. If we have wanted to ensure that all three customers buy contracts, we would have needed to set $a = 135/3 = 45$ and this would have produced revenue of only \$ 270.

Now suppose we allow bookings to be made just one week ahead. It will be optimal to take $a = 100$ and $b = 180$. All customers will make purchases and the revenue will be $100 + 190 + 136 = \$426$. Note that we are requiring customer 1 to pay \$ 10 more than before. If we wished to avoid this we might take $a = 95$ and $b = 190$. Now we can be certain that customers 1 and 2 will purchase the same as before, and the revenue will be $95 + 190 + 133 = \$418$.

The nice thing about this last solution is that we need not know before introducing the new tariff that the third customer even exists. If he does, then we increase revenue by 46.7%. But if he does not exist then nothing has been lost. If we are lucky, there will be even further customers, additional to customer 3, who will start buying contracts.

This example suggests that a provider may increase its profit by using a mixed contract and that customers may obtain at least the same net benefit they obtained in the case of using optimal static contracts. But what if the provider uses purely dynamic contracts? In this case the optimal revenue is \$370, for $b = 100$, which is larger than the optimal static revenue of \$285. But this is not always the case! If we have a different percentage of the same customers, the optimal static revenue may be higher. For instance, if no customer 3 is present, then the maximum revenue of static, dynamic and mixed contracts are \$285 ($a = 95$), \$250 ($b = 100$) and \$290 ($a = 100$, $b = 180$) respectively. Hence dynamic contracts may no longer be optimal for the provider.

This paper is organized as follows. In Section II, we formulate the general optimization problem that a customer faces when provided with a static, mixed or dynamic contract. Section III defines the utility function that will be used throughout the rest of the paper and in Section IV, we use this utility function to model the customer's net benefit maximization problem. In Section V, we model the provider's revenue maximization problem and we show how the prices of the various contracts are formed. We also consider various cases of customer distributions. Section VI models a price competition game between providers of different type of contracts, where we give an insight of how prices will adjust. One can prove that, under certain conditions, at the equilibrium the market will be segmented between the providers. Section VII concludes our work and provides some points for future work.

II. The optimization problem

Let us start from the position that contracts generate charges over fixed long time periods consisting of n shorter periods (slots). Prices are assumed to be fixed and known to customers. They buy long-term contracts where they secure a certain fixed amount of resources for the n slots, and may combine these with short-term contracts for acquiring additional resources in each slot.

For simplicity assume that the long time period is a year and slots correspond to weeks, i.e., $n = 52$. There are N customers who are prepared to buy long-term contracts given that we are charging a static price corresponding to $\$a_0$ per

1 Mbps per week (the actual price is na_0 for the long-term period but we like to express it per slot). Customer i buys bandwidth of x_i Mbps in this long-term static contract. The revenue is presently

$$r(a_0, b = \infty) = na_0 \sum_{i=1}^{N} x_i \, .$$

Suppose now that we introduce the possibility of buying further units of bandwidth, one-week-ahead, for $\$b$ per 1 Mbps. Suppose that for the coming week, customer i has a utility for bandwidth that is parameterized by θ, which he can predict one week ahead. If he already owns a year-long contract for bandwidth x, and he additionally buys bandwidth y for the coming week, then his utility is $u_i(\theta, x + y)$. This customer will choose to buy a year-long contract for x_i where this comes from his finding his maximum net benefit as

$$nb_i(a, b) = \max_{x_i} \left\{ nE\Big[\max_{y_i} \{u_i(\theta, x_i + y_i) - by_i\} \Big] - nax_i \right\}, \tag{1}$$

where the expectation is taken over θ and $n = 52$.

Observe first that the solution of the optimization problem is independent on n when both prices are expressed on a per slot basis. The average revenue he generates is $r_i(a, b) = nax_i + nb\bar{y}_i$, where $\bar{y}_i = E\left[\arg\max_{y_i} \{u_i(\theta, x_i + y_i) - by_i\}\right]$ is the average bandwidth he will buy in a week in the dynamic bandwidth market, and x_i is the maximizer in (1). The total revenue may be denoted as $r(a, b) = \sum_i r_i(a, b)$. We seek to maximize $r(a, b)$.

The problem faced by the supplier offering the tariff (a, b) becomes more complex if we consider the possibility that the customer may seek another supplier offering the tariff (a_1, b_1) if his net benefit using this supplier is higher. For instance, suppose the second supplier offers a purely static contract (a_0, ∞). This suggests for a market consisting of a single customer the optimization problem

$$\underset{a,b}{\text{maximize}} \; r(a, b) \; \text{s.t.} \quad nb(a, b) \geq nb(a_0, \infty) \, . \tag{2}$$

Intuition suggests that at optimality, $a \leq a_0$ and $b > a$. We also observe that for customers with constant bandwidth requirements introducing the possibility of buying bandwidth on demand does not increase the revenue of the supplier. But in the case of customers with fluctuating requirements, (2) should lead to an increase of revenue.

We can derive now an important property of static versus purely dynamic contracts. Under the same price $a = b$, dynamic contracts are more beneficial for the customer. This is easy to see from (1) since

$$\max_x \Big\{ nE\Big[u_i(\theta, x) - ax \Big] \Big\} \leq nE\Big[\max_y \{u_i(\theta, y) - ay\} \Big] \, .$$

A simpler version of the problem is one in which customers have fluctuating but predictable bandwidth requirements. This corresponds to their knowing the realization of the values of θ for the different weeks in advance. In this case the customer solves

$$nb_i(a, b) = \max_{x_i} \left\{ \sum_{j=1}^{n} \Big[\max_{y_i^j} \{u_i(\theta_j, x_i + y_i^j) - by_i^j\} \Big] - nax_i \right\}, \tag{3}$$

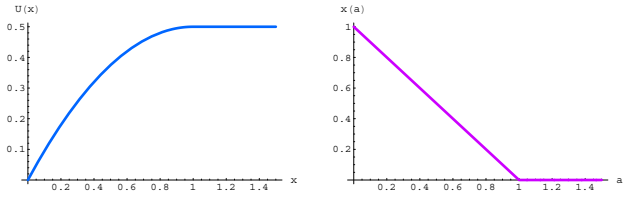


Fig. 1. The utility function and resulting demand function when $k = 1$.

where y_i^j is the bandwidth to be bought from the dynamic market during the jth period. Now the revenue generated is $r_i(a, b) = nax_i + b\sum_j^n y_i^j$.

This model allows us to formulate some more interesting problems. Observe that the dynamic demand y_i^j of customer fluctuates over time and its statistics are affected by the choice of a, b. If there is finite capacity in the system that must be procured at the beginning of the year, then we must also make sure that the total demand from customers rarely exceed the capacity. This suggests that we associate an effective bandwidth [7] with each customer i capturing the variability of y_i^j. Lets denote this by $eb_i(a, b)$. Then a related optimization problem is

$$\underset{a,b}{\text{maximize}} \; r(a, b) \quad \text{s.t.} \quad \sum_i eb_i(a, b) \leq C \, . \tag{4}$$

This captures the fact that increasing a and decreasing b may result in higher revenues but it creates more fluctuating demand which may be harder to provide.

III. A Model for Consumer Utility

We suppose that at the time that a customer purchases static bandwidth he does not yet know his utility function for bandwidth. He knows only that is will be of the form $u_k(x)$, where k is a parameter, presently unknown, but distributed a priori as a random variable with a known distribution function $F(k)$. In particular, we will illustrate ideas with

$$u_k(x) = \begin{cases} kx - \frac{1}{2}x^2, & x \leq k \, , \\ \frac{1}{2}k^2, & x \geq k \, . \end{cases} \tag{5}$$

Assuming this utility function, if the user faces a static price of a and knows k then he will choose x to maximize $u_k(x) - ax$. This gives the demand function $x_k(a) = \max\{k - a, 0\}$ (See Fig. 1).

IV. The Consumer's Problem: Maximizing Net Benefit

Consider a single consumer and the purchases of bandwidth that he will make when faced with a static, dynamic, or mixed supplier, and where the static and dynamic bandwidth prices are a and b respectively. If facing a static, dynamic or mixed supplier, the customer optimizes over the quantity of bandwidth he buys and obtains net benefits (average, per slot)

of

$$nb_S(a) = \max_x \Big\{ E\left[u_k(x) - ax\right] \Big\},$$
$$nb_D(b) = E\left[\max_x \{u_k(x) - bx\}\right],$$
$$nb_M(a,b) = \max_x \left\{ E\left[\max_y \{u_k(x+y) - by\} - ax\right] \right\},$$

where the expected values are taken over k.

To illustrate the sort of things that can occur, we consider examples in which the utility function is of the form (5) and k has some special distributions. One of these is where k is arbitrarily distributed on $[0,1]$ (see Section V-A). Let us consider for the moment only the cases of static and dynamic sellers (not mixed). We then have,

$$nb_S(a) = \max_x \left\{ \int_0^1 u_k(x)\, dk - ax \right\},$$
$$nb_D(b) = \int_0^1 \max_x \big\{ u_k(x) - bx \big\}\, dk\,.$$

Another possible distribution for k is when $k = k_1$ or $k = k_2$ with probabilities $1-p$ and p respectively (see Section V-B). We suppose $0 \le k_1 < k_2$. We now have

$$nb_S(a) = \max_x \Big[(1-p)u_{k_1}(x) + pu_{k_2}(x) - ax \Big]\,.$$

For a fixed a, this is a convex function of p. For dynamic contracts the user's net benefit is

$$nb_D(b) = (1-p) \max_x \big[u_{k_1}(x) - bx \big] + p \max_x \big[u_{k_2}(x) - bx \big]\,.$$

For a fixed b, this is a linear function of p. Thus, there will be some range $p \in [p_1, p_2] \subset [0,1]$ for which the dynamic contract will be preferred. To see this, note that if $a = b$ then the user will prefer to buy dynamically, except at $p = 0$ and $p = 1$, where the user is indifferent. As b increases, $nb_D(b)$ decreases at every value of p and remains a linear function of p. Thus as functions of p the linear function $nb_D(b)$ crosses the convex function $nb_S(a)$ at most twice.

V. The monopolist supplier's problem

A. Identical customers with arbitrarily distributed k

Suppose there is a single monopolist supplier who can supply bandwidth at cost c and who is attempting to sell to a population of identical customers. Let us ask whether he can make more profit as a static, dynamic or mixed supplier. To discover the answer, we must perform a somewhat complicated calculation. We assume that the supplier knows F, the distribution of k and can determine a typical customer's demand function, say $x_S(a)$ in the static case. He then chooses the price a to maximize the profit $(a-c)x_S(a)$.

The results of Table I are obtained numerically under the assumption that k is uniformly distributed on $[0,1]$. We suppose $c = 0$ so that the supplier's revenue and profit are the same thing. We show what happens for the three types of supplier in the columns labeled (S), (M) and (D). The figures for revenue and mean bandwidths bought and sold are per customer.

TABLE I

(S) The optimal purely static contract has $a = 2/9$ and the optimal revenue is $2/27$. The user buys $x = 1/3$.The user's net benefit is $7/162$. The mean bandwidth consumed is $1/3$. **(M)** The optimal mixed contract has $a = 6/25$ and $b = 2/5$ and optimal revenue $2/25$. The user buys $x = 1/5$ static and his net benefit is $11/250$. The mean bandwidth consumed is $7/25$. **(D)** The optimal purely dynamic contract has $b = 1/3$ and optimal revenue $2/27$. The user's net benefit is $4/81$. The mean bandwidth consumed is $2/9$.

	(S)	(M)	(D)
seller revenue	0.0741	0.0800	0.0741
optimal a	0.2222	0.2400	
optimal b		0.4000	0.3333
bandwidth bought in static	0.3333	0.2000	
mean bandwidth sold	0.3333	0.2800	0.2222
user net benefit	0.0432	0.0440	0.0494

These numerical results suggest some conjectures. If true in general, they support the notion that there is advantage in offering mixed contracts.

A. The revenue achieved by the seller is strictly greater in (M) than in (S) or (D).
B. The mean bandwidth sold decreases from (S) to (M) to (D).
C. The user's average net benefit increases from (S) to (M) to (D).

We also notice the interesting fact that the revenue achieved by the seller is the same in (S) and (D). This turns out to be true in general, assuming the utility is of form (5). We state this as follows.

Proposition 1: Suppose the utility function is of the form in (5). Then a seller of static contracts who maximizes his revenue by choice of a obtains the same revenue as a seller of dynamic contracts who maximizes his revenue by choice of b. Moreover, at these optimums the bandwidth x^* that is sold by the seller of static contracts is equal to the optimum dynamic contract price price b^*.

Proof: The theorem holds whatever the distribution of k, but let us suppose for simplicity that k is arbitrarily distributed over $[0,1]$ with density function $f(k)$. If the seller chooses a price a for static contracts, the buyer will choose x to maximize

$$\int_0^x u_k(k) f(k)\, dk + \int_x^1 u_k(x) f(k)\, dk - ax\,.$$

This is maximized where

$$-a + \int_x^1 (k-x) f(k)\, dk = 0\,.$$

Thus the seller maximizes his revenue by maximizing over x

$$x \int_x^1 (k-x) f(k)\, dk\,. \tag{6}$$

Now if the price of dynamic contracts is b, then the amount of bandwidth purchased is $(k-b)^+$. Thus the seller seeks to maximize over b

$$b\int_b^1 (k-b)f(k)\,dk\,. \tag{7}$$

The theorem follows by comparing (6) and (7). ■

Note also that the optimal x is where

$$\frac{d}{dx}\left\{x\int_x^1 (k-x)f(k)\,dk\right\} = \int_x^1 (k-2x)f(k)\,dk = 0\,.$$

So taking $x = x^*$ as the solution,

$$\int_{x^*}^1 (k-x^*)f(k)\,dk = (1-F(x^*))\,x^* < x^*\,.$$

Thus as $x^* = b^*$ we have that the bandwidth sold under the dynamic contract is less than the bandwidth sold under the static contract, as in Conjecture B.

B. Identical customers with two-point distributed k

Since we would like to consider what happens when customers are not identical we wish to find some simple model in which their differences are captured by a single parameter. So let us suppose that $k = k_1$ or $k = k_2$ with probabilities $1-p$ and p. Now different values of p can distinguish customers from one another. Initially, let us fix p and consider what happens when a monopolist is selling to a population of customers, all of whom have the same p.

1) Static contracts: The customer's net benefit is

$$p_1 u_{k_1}(x) + p_2 u_{k_2}(x) - ax\,.$$

This is a continuous function of x, and it has a continuous first derivative. Taking the first derivative, we see that the price at which the user will purchase x is

$$p_1 u'_{k_1}(x) + p_2 u'_{k_2}(x) = p_1(k_1-x)^+ + p_2(k_2-x)^+\,.$$

In fact, we can write

$$p_1(k_1-x)^+ + p_2(k_2-x)^+ = \max\Big\{0, \bar{k}-x, p_2(k_2-x)\Big\}\,.$$

Or we can also invert this to write the demand function as

$$x(a) = \max\Big\{0, \bar{k}-a, k_2 - a/p_2\Big\}\,.$$

Thus the seller's maximum revenue is

$$\begin{aligned}
&\max_x x\max\Big\{0, \bar{k}-x, p_2(k_2-x)\Big\}\\
&= \max\Big\{0, \max_x\{x(\bar{k}-x)\}, p_2\max_x\{x(k_2-x)\}\Big\}\\
&= \max\Big\{\tfrac{1}{4}\bar{k}^2, \tfrac{1}{4}p_2k_2^2\Big\}
\end{aligned}$$

Now $\frac{1}{4}\bar{k}^2 \geq \frac{1}{4}p_2k_2^2$ if and only if $p_2 \leq p^*$ where

$$p^* = \frac{1}{(k_2/k_1-1)^2}\,,$$

since $p_1 = 1-p_2$. This gives the following.

Proposition 2: The optimal (a^*, x^*) are either $(\frac{1}{2}\bar{k}, \frac{1}{2}\bar{k})$ or $(\frac{1}{2}p_2k_2, \frac{1}{2}k_2)$ as p_2 is less or greater than p^*.

Note that $2k_1 \geq k_2 \implies p^* \geq 1 \implies p_2 \leq p^*$ and the optimum is at $x^* = \frac{1}{2}\bar{k}$ for all values of p_2. Conversely, $p_2 > p^* \implies 2k_1 < k_2$.

2) Dynamic contracts: If the seller sets price b then the user will buy average bandwidth of

$$p_1(k_1-b)^+ + p_2(k_2-b)^+\,.$$

We know by Proposition 1 that the optimal value of b is $\frac{1}{2}\bar{k}$ or $\frac{1}{2}k_2$. If $p_2 \leq p^*$ we have $b = \frac{1}{2}\bar{k}$. If $p_2 \geq p^*$ we have $b = \frac{1}{2}k_2 > k_1$. It turns out that in both cases the average bandwidth is the same as in the case of the static contract. Hence, Conjecture B is not true in a strict sense for the case of the two-point distributed k. But under a arbitrarily distributed k, Conjecture B is true in a strict sense (as shown in Section V-A).

Furthermore, we can compute the difference in the user's net benefit under optimal purely dynamic and purely static contracts as

$$nb_D - nb_S = \begin{cases} \frac{1}{2}p_1p_2(k_2-k_1)^2, & p_2 \leq p^*\\ -\frac{1}{2}p_1k_1^2, & p_2 > p^* \end{cases}$$

This gives

Proposition 3: If $p_2 > p^*$ the a user strictly prefers to face a monopolist seller of static contracts rather than a monopolist supplier of dynamic contracts. If $p_2 < p^*$ his preference is reversed.

Thus, we see that whereas the optimal purely static and purely dynamic contracts produce the same revenue for the seller, the type of contract that obtains the greater net benefit for the user depends on whether p_2 is less or greater than p^*. Notice that there is a discontinuity in the user's net benefit at $p_2 = p^*$. Conjecture C is not true in general. It is only true if $p_2 \leq p^*$.

Remark: The above proposition can be generalized, in the sense that for every price a of the static provider, a dynamic provider can publish a price b that offers him at least the same revenue, while the customer acquires a higher net benefit, if $p_2 \leq p^*$. The result is reversed in the case of $p_2 > p^*$. In Fig. 2 we show how the user's net benefit can vary with the seller's revenue as the seller varies his static price a, or dynamic price b. The sellers have no cost. Notice that the maximum revenue that can be obtained by either method is the same.

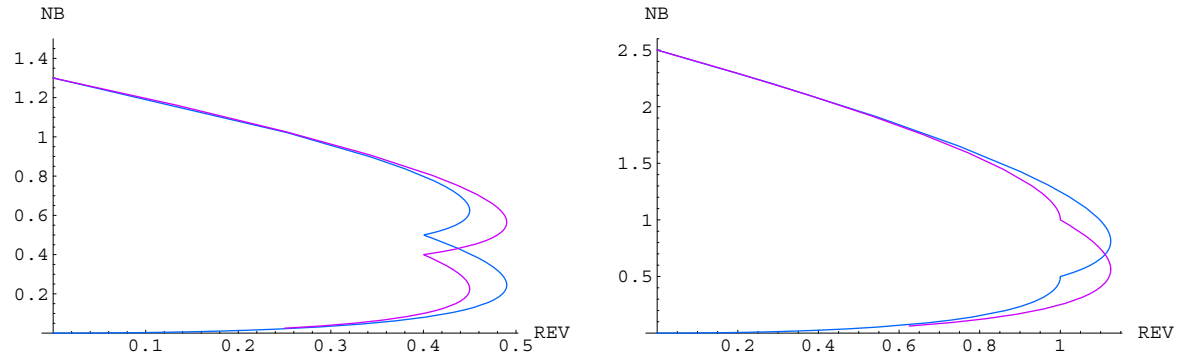
3) Mixed contracts: Consider the optimal mixed strategy, optimized over prices $a \leq b$. Suppose the net benefit at the optimum is

$$p_1 u_{k_1}(x+y_1) + p_2 u_{k_2}(x+y_2) - xa - p_1y_1b - p_2y_2b\,.$$

Note first that we cannot have $y_1 > 0$, since as $b > a$ the net benefit to the user could be improved by taking $x \to x+y_1$, $y_1 \to 0$ and $y_2 \to y_2 - y_1$. Therefore we may consider the problem when the net benefit is of the form

$$p_1 u_{k_1}(x) + p_2 u_{k_2}(x+y_2) - xa - p_2y_2b\,.$$

Now we must have $p_2b < a$, otherwise the user does best to buy only static. Now also note that we cannot have $x > k_1$. For if this were so, the user could make the change $x \to x-\epsilon > k_1$ and $y_2 \to y_2+\epsilon$, and his net benefit would increase by $(a-p_2b)\epsilon > 0$. Thus the optimum must occur where $x \leq k_1$. Note also, that the optimum must occur where $x + y_2 \leq k_2$.



(a) $p_2 = 0.2$ (b) $p_2 = 0.5$

Fig. 2. The user's net benefit (NB) against the seller's revenue (REV). The static seller is shown in blue and the dynamic seller in violet. In this example, $k_1 = 1$, $k_2 = 3$. If $p_2 < p^* = 0.25$, the seller of dynamic (violet) can, at any revenue value, offer the user a greater net benefit than can the seller of static (blue). The reverse is true if $p_2 > p^*$.

For a given x, such that $x < k_2 - b$, the user optimally takes $y_2 = (k_2 - x - b)$, and so his net benefit is then

$$p_1 u_{k_1}(x) + p_2 u_{k_2}(k_2 - b) - xa - p_2(k_2 - x - b)b\,.$$

The derivative of this with respect to x is

$$-a + p_2 b + p_1(k_1 - x)\,.$$

Since $-a + p_2 b < 0$, this is negative at $x = k_1$, and so the optimum is at some $x < k_1$, where $x = k_1 - b + (b - a)/p_1$. Thus the seller's revenue can be found by substituting this value of x into his revenue of $xa + p_2(k_2 - x - b)b$ and optimizing over a and b. This gives

$$a^* = \tfrac{1}{2}\bar{k} \quad \text{and} \quad b^* = \tfrac{1}{2}k_2\,.$$

Note that these satisfy $b > a > p_2 b$, as required. The optimal purchases are $x = \frac{1}{2}k_1$ and $y_2 = \frac{1}{2}(k_2 - k_1)$. Thus the mean amount that is purchased is $x + p_2 y_2 = \frac{1}{2}\bar{k}$. Interestingly, we have the same static price, and same average bandwidth purchased, as in the static contracts case, provided we are in the regime where $p_2 \le p^*$. As commented previously, a special case is $2k_1 \ge k_2$, which implies $p^* = 0$.

The user's net benefit can be calculated to be

$$p_1 u_{k_1}(k_1/2) + p_2 u_{k_2}(k_2/2) - (1/4)\big(p_1 k_1^2 + p_2 k_2^2\big) = \tfrac{1}{8}\big(p_1 k_1^2 + p_2 k_2^2\big)$$

Under the static contract it was

$$\tfrac{1}{8}\big(p_1 k_1 + p_2 k_2\big)^2 = \tfrac{1}{8}\bar{k}^2\,,$$

which is less. The seller's revenue has increased from $\frac{1}{4}\bar{k}^2$ to $\frac{1}{4}(p_1 k_1^2 + p_2 k_2^2)$. It is interesting to note that both the seller and user increase their profit and net benefit, respectively, by the same factor of

$$\frac{p_1 k_1^2 + p_2 k_2^2}{(p_1 k_1 + p_2 k_2)^2}\,.$$

The greatest possible value for this ratio in the region $2k_1 \ge k_2$ is 1.125.

TABLE II
SUMMARY OF RESULTS

$p_2 \le p^*$	$p_2 > p^*$
$r_S = r_D < r_M$	$r_S = r_D < r_M$
$nb_S < nb_M < nb_D$	$nb_D < nb_S < nb_M$
$m_S = m_M = m_D$	$m_S = m_D > m_M$

Remark: Note that the seller obtains the same revenue as he could obtain if he knew k and used 'personalized pricing', i.e., charging a price $\rho = \frac{1}{2}k_1$ if $k = k_1$, and $\rho = \frac{1}{2}k_2$ if $k = k_2$.

Let us summarize results. In Table II, we write the seller's revenue as r_S, r_M, r_D, the user net benefits as nb_S, nb_M, nb_D and the mean bandwidths sold as m_S, m_M, m_D.

These results suggest that mixed contracts are always preferable to static contracts, in the sense that $r_M > r_S$, $nb_M > nb_S$ and $m_M \le m_S$.

VI. Price competition amongst suppliers

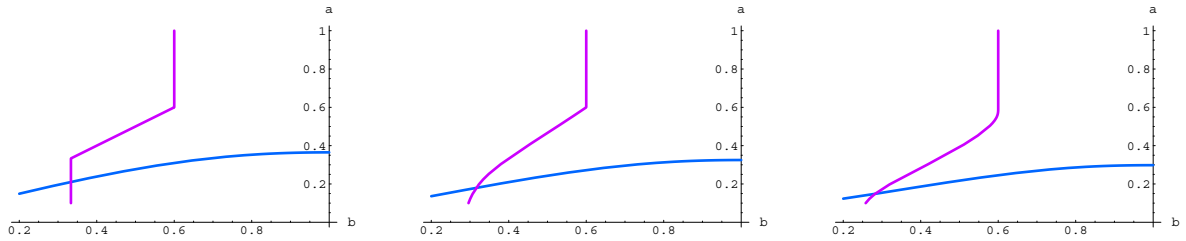
We now turn to investigate what happens when suppliers compete with one another on price. Who wins? Is it the supplier of static, dynamic or mixed contracts? Is there an equilibrium in which suppliers of different types can both make positive profit?

Let us suppose that for a given customer $k = 0$ or $k = k_2$, with probabilities $1 - p$ and p. We suppose $p \in [0, 1]$ is distributed across the population of users with a density function of $f(p)$.

As previously, the static provider sells at price a, but the contract for purchase must be made before the customer knows whether his k is equal to 0 or 1. The dynamic provider sells at price b, and with the flexibility that the customer need not make the purchase until he knows whether his k is equal to 0 or 1.

If buying from the static provider the customer chooses x to maximize

$$pu(1, x) - ax\,,$$



(a) (b) (c)

Fig. 3. The reaction curves $a(b)$ against b (in blue), and a against $b(a)$ (in violet), for different distributions of p.

and so optimally buys $\max\{1 - a/p, 0\}$. If buying from the dynamic provider the customer chooses x to maximize

$$pu(1, x) - pbx\,,$$

and optimally buys $\max\{1 - b, 0\}$. Thus a customer strictly prefers to buy from static rather than dynamic if and only if $a < pb$.

Suppose the two providers have unit cost of c_1 and c_2, respectively. Then the profits obtained by the two providers are

$$f_A(a,b) = \begin{cases} (a - c_1) \int_{a/b}^{1} (1 - a/p) f(p)\, dp, & a < b \\ 0, & a \geq b \end{cases}$$

$$f_B(a,b) = \begin{cases} (b - c_2) \int_{0}^{a/b} p(k - b) f(p)\, dp, & a < b \\ (b - c_2) \int_{0}^{1} p(k - b) f(p)\, dp, & a \geq b \end{cases}$$

These are the payoffs in a Bertrand game of price competition. From these we can compute the reaction curves.

$$a(b) = \arg\max_{a}\{f_A(a,b)\}, \qquad b(a) = \arg\max_{b}\{f_B(a,b)\}\,.$$

To give some numerical examples, suppose $c_1 = 0.1$ and $c_2 = 0.2$. Fig. 3(a) shows the reaction curves when p is uniformly distributed, i.e., $f(p) = 1$. Fig. 3(b) shows these curves when $f(p) = 6p(1 - p)$. Fig. 3(c) shows these curves when the distribution is more concentrated around $p = 1/2$, with $f(p) = 630p^4(1 - p)^4$. The point of intersection in each graph is a Nash equilibrium. In the equilibrium point of Fig. 3(a), $a = 0.2105$, $b = 0.3333$ and the respective revenues are 0.0300 and 0.0177.

As already mentioned in Section IV, at the equilibrium point the market will be segmented between the two providers. The customers that reside in the edges of the p's distribution will prefer the static provider and the rest customers will prefer the dynamic provider.

VII. Conclusions & Future work

We have presented a model for analyzing the different types of a contract in a market of resources that can be provisioned on demand. Our aim was to show that, when using the appropriate pricing scheme, the provider's goal of revenue maximization is not opposed to the consumer's goal for net benefit maximization. We have captured the fluctuating demand of the customers with a single parameter (p), leading to a formulation of high and low demand periods. Based on this model, we have studied the characteristics of such contracts, under specific distributions of k. We have seen how the provider's revenue, the customer's net benefit and the mean bandwidth sold vary from one type of contract to another and we have provided the conditions under with a mixed contract is better from the rest for both the provider and the consumer. We have also proved that the optimal revenue of a static provider is the same with the revenue achieved by a dynamic provider, under any distribution of k. Finally, we have provided some numerical results for a price competition game between a static and a dynamic provider that shows that a Nash equilibrium point exists and that the market at this point is segmented, hence both providers make profits.

Many issues are open for further research. An interesting point is to see what segment of the market is obtained by a mixed provider when participating in a price competition game with a static or dynamic provider. Furthermore, apart from the example with the two-point distribution of k, results from more generic distributions will be studied. The case of non-identical customers needs also to be studied. Extensions with effective bandwidth will also be considered. An important question that may rise is how the shape of the utility function affects all the aforementioned results. Finally, an open issue is how such a model can be extended for Grid resources, since the definition of the basic resource in Grids and its characteristics are not yet well-defined.

References

[1] "DSL Forum's BroadbandSuite: Unleashing the Power of Broadband," Technical Paper, DSLForum.org.

[2] "Creating a Commercially Sustainable Framework for IP Services Realizing Next Generation Revenues," White paper, IPSphere Forum, May 2006.

[3] "GRID COMPUTING on SUN: Deploying Scalable and Efficient HPC Grids in the Modern Commercial or Technical Enterprise," White Paper, SUN Microsystems Inc., March 2006.

[4] B.M. Mitchell and I. Vogelsang, "Telecommunications Pricing Theory and Practice," *Cambridge University Press*, 1991.

[5] T. Bergstrom and J.K. MacKie-Mason, "Some Simple Analytics of Peal-Load Pricing," *The RAND Journal of Economics*, vol. 22(2), pp. 241-249, 1991.

[6] W.Y. Oi, "A Disneyland Dilemma: Two-part Tariffs for a Mickey Mouse Monopoly," *Quarterly Journal of Economics*, vol. 85, pp. 77-96, Feb. 1971

[7] F.P. Kelly, "Notes on effective bandwidth," In: F. Kelly, S. Zachary and I. Ziedins (Eds.), "Stochastic Networks: Theory and Applications in Telecommunication Networks," Volume 4 of *Royal Statistical Society Lecture Notes Series*, Oxford, pp. 141-168, Oxford University Press, 1996.

Adopting the Grid for Business Purposes: The Main Objectives and the Associated Economic Issues

George A. Thanos[1], Costas Courcoubetis[1] and George D. Stamoulis[1]

[1]Network Economics and Services Laboratory, Department of Informatics
Athens University of Economics and Business, Patision 76, Athens 104 34, Greece
{gthanos, courcou, gstamoul}@aueb.gr

Abstract. Grid technology offers numerous opportunities for the players involved. Despite the fact that the academic community has already exploited many of them, there is an evident reluctance from the business community to act likewise. Recent analysis reveals that the problem lies in overcoming certain business barriers rather than technological ones. At this stage understanding the real-life economic issues from a business perspective is deemed as more important than gaining understanding of complex theoretical economical problems, such as those related to accounting or resource sharing mechanisms especially in cases where the players do not exhibit the required technological expertise. This paper is stimulated from interaction with players from the industry and aims to fill this gap. In particular, we identify and evaluate a number of economic issues that should be taken into consideration by industrial players so that their trust and confidence in the adoption of this promising technology be increased.[1]

Keywords: Grid Technology, economics, resource sharing, virtual organisations, market, economies of scale, network externalities etc

1 Introduction

Grid technology promises a new way of delivering services across IP-based infrastructures. These range from common ones, such as existing mass multimedia services, to more complex and demanding customised industrial applications. Over the last years Grid technology has proven its merits through enabling the execution of highly resource demanding applications for the scientific community some of which were previously only realised over expensive high-performance computing (HPC) centres.

However, in order for Grid technology to fulfil the aforementioned promise, it has first to be adopted by the diverse business community thus being provided and consequently validated, by a significantly larger number of providers and users. Recent studies [1] and European initiatives [2] have indicated a reluctance and slow take-off of Grid technology and market by the industry, something attributed mainly to economic and market barriers rather than to technological ones.

[1] This project has been partly supported by FP6 EU-funded IST projects BEinGRID (IST5-034702) and GridEcon (IST5-033634)

So far there has being a lot of work around theoretical economical analysis examining issues like accounting and resource sharing mechanisms for Grid architectures etc. However, our experience from interacting with industry players and discussing their concerns has shown that prior to solving complex architectural issues there is an evident need for analysing the Grid phenomenon and its economic side effects from a business perspective. Thus, in this paper we identify and analyse those criteria and economic issues that a new player should take into consideration prior to making his decision whether to adopt Grid technology for his business or not and how these will affect his Grid business afterwards. Such a decision can be made by means of a relevant model, which will take the factors identified in the present paper as inputs. Our overall aim is to increase the confidence of the industry towards Grid adoption by exposing the business issues, both positive and negative ones, that once taken into careful consideration by the value chain players will enable them to realise the numerous opportunities that Grid technology has to offer and at the same time construct feasible business plans to fully exploit them.

Our identification and analysis has been performed with support by the Integrated Project Business Experiments in Grid – BEinGRID [3], European Union's largest integrated project funded by the Information Society Technologies (IST) research, part of EU's Framework Programme 6 [4]. The communication with 18 real-life Business Experiments from various industries provided the practical framework to validate our theoretical analysis.

The paper is structured as follows: a brief introduction to Grid economics is presented in Section 2 followed by a discussion on the main economic objectives for adopting the Grid and an initial identification of associated economic issues in Section 3. Section 4 identifies and analyses a number of economic issues related to Grid adoption from the industry whereas Section 5 provides a case study and evaluation of how these issues affect real-life scenarios. Section 6 provides some concluding remarks.

2 A Brief Introduction to Grid Economics and Related Work

Firstly, it is imperative to review some basic definitions related to Grid Technology and the current work in Grid Economics. To start with, we define a Grid service as a Web Service that provides some well-defined interfaces and follows specific conventions [5]. The interfaces address issues such as address discovery, dynamic service creation, lifetime management, notification, and manageability. The conventions regulate naming and upgradeability of services. Each service described in the Open Grid Services Architecture (OGSA) [6] is a single Grid service or a composition of Grid services. A Grid middleware is typically composed of several Grid services with different functionality. Usually, at least the following functionalities are covered: resource management, Job management, Service discovery, scheduling, accounting and security.

Nowadays, a single business process and value chain of a company can be performed by several business partners. The company involved in this process is then a virtual company or organization (VO), as it is only a temporary aggregation of partners in order to perform a specific process. The corresponding concept from economics is called the coalition. VOs can be seen one of the most important drives

for Grid technology adoption as it allows these organisations to efficiently share and utilise their geographically distributed computing, storage and data resources over a common infrastructure.

Among the first to raise a number of true economic issues focused on the commercialization of Grid resources (i.e. computing) were Kenyon and Cheliotis. Specifically, in their work they argue that Grid commodity is rather a stochastic one rather than as a deterministic one (such as oil, electricity, etc). Since Grid resources are non-storable, the authors claim that future contracts will be the basic building blocks in Grid environments instead of spot markets. Market uncertainty and decision support are the most important issues that need to be addressed in this context.

The authors identify a set of requirements for commercialization of Grid resources such as product construction and reservation, contract management, clearing, accounting and billing, trading support, price formation and decision support. Also, in [7], Cheliotis et al. set a number of important questions on the successful creation of a Grid market. They argue that the most important part for a successful Grid market creation is to fully understand and foster user requirements and demands. Overall, [8], [9], [7] mostly define the most important issues for Grid commercialization and they do not propose any specific solutions for them.

Gray on the other hand in [10] discusses the economic tradeoffs of doing Grid-scale distributed computing (WAN rather than LAN clusters). Specifically, Gray analyzes the economics of outsourcing. Using simple commercial examples, he calculates the corresponding value of 1$ for bandwidth over the WAN, for number of CPU instructions, for CPU time, for disk space, for database accesses and for disk bandwidth. Identifying communication cost as a bottleneck, Gray concludes on a rule of thumb regarding outsourcing, according to which computations must be nearly stateless and have more than 10 hours of CPU time per GB of network traffic before outsourcing the computation makes economic sense. Otherwise, LAN cluster provide a more economically viable alternative.

Probably the most extensive work on Grid Economics up-to-date has been performed by the GRIDS (Grid Computing and Distributed Systems) laboratory, headed by Buyya. Their most significant research work related to our work is the Economy Grid project where it is clearly identified that a major challenge for next-generation Grid computing is the creation of an “Economy Grid”, meaning a competitive realistic Grid Marketplace that regulates supply and demand, and offers the right incentives to players (suppliers and consumers) for improving the utilization of resources. The next step was the Gridbus [11] project, aiming at producing a set of Grid middleware technologies to support e-science and e-business applications. In some of the designed and developed components for this technology one will find incorporated features relevant to “Grid Economics”, such as a broker agent software for job scheduling, a market directory for publishing and searching for available services, and a centralized infrastructure that provides accounting and payment services. The “Economy Grid” project, the GRACE architecture and an overview of related work on price setting, market-based resource allocation and scheduling systems are presented in [12].

Other works in the Grid Economics include a centralized strategy-proof architecture for Grid Computing by Egg [13] and the Mojo Economy [14], the Weng

Price-setting mechanisms [15], the price prediction mechanisms by [16], and work driven from European funded IST projects.

As already mentioned the aforementioned work is more focused in the theoretical analysis of economic mechanisms and fails to analyse specific economic issues from a business perspective such as the economies of scale/scope, network externalities, free-riding problems, information asymmetry, and impacts to other markets etc which we will address in the subsequent sections.

3 Economic Objectives for Adopting the Grid and Initial Identification of the Associated Economic Issues

The aim of this section is to discuss the main economic objectives for adopting the Grid for Business and identify the underlying economic issues/concerns. We propose at this stage the main three alternative economic reasons for Grid to be used in commercial applications. By keeping the number of alternatives small and hence abstracting to a significant level the implementation context, we can understand the economics better. These are discussed in the subsequent sections.

3.1 Optimization of Processing Power in a Single Organization

A single organization may require processing power that cannot be provided by means of stand-alone machines. By interconnecting these machines in a Grid, high processing power can be used even by a single application. Thus, the organization achieves both a high peak processing capacity and a high average utilization of the processing power available, since this can be flexibly allocated to multiple Grid-enabled applications. These features also lead to increased cost-efficiency for the infrastructure deployed. This is particularly important for a large organization with several departments scattered around the world, each possessing its own local infrastructure. Interconnecting these in a Grid attains the aforementioned performance enhancement, high exploitation of resources, and cost-efficiency and economies of scale, due to the fact that interconnection of all machines improves utilization of each individual one. Moreover, the whole approach is scalable, due to the fact that the Grid middleware provides automatic load balancing and transparent usage of the hardware. Besides these, if the various departments possess complementary infrastructure, then the organization also attains economies of scope.

Regarding management, since the Grid belongs to single organization, a centralized approach is always an option. On the other hand, particularly if there are multiple departments in the organization, with some notion of autonomy (e.g. own infrastructure contributed to Grid and IT budget), then self-management of the Grid by means of economic/market mechanisms is possible and probably preferable. Indeed, the centralized approach requires complete information, which is not always straightforward to gather in a highly distributed single-domain system. On the other hand, a market mechanism defining prices for accessing and using the Grid resources by the various departments provides the right incentives for rational usage and results in shaping of demand according to the actual needs, which in fact may be thus discovered; prices may either be really monetary, or virtual ones with each department being allocated a Grid virtual budget. This approach also requires

accounting functionality, e.g. for monitoring the usage of resources by the various departments and assigning the relevant charges, as well as specification of the right SLAs and appropriate tariffs for them.

3.2 Sharing of Complementary Resources in Multi-provider Environments

Consider a group of organizations, each of which possesses its own resources, which are complementary to each other. For example, organization A possesses a powerful database server, while B has a huge amount of data and C possesses an application running over its server that requires data such as that of B. Clearly, when collaborating in the form of Grid, all organizations can bring together a powerful outcome, while each of them exploits very highly its own resources at a cost-efficient way, without needing to invest to the missing resources that are now contributed by others. In this case, the collaborating organizations enjoy economies of scope, since bringing all resources together by means of Grid broadens their scope of applicability. Apart from serving their own needs by forming a Grid, organizations with complementary resources may also form a Virtual Organization serving third parties. The formation of VOs has a considerable impact to the market; see item 3 below. If the group forming the Grid is not closed, then network externalities and economies of scale may arise in the case where new organizations can join the group, thus enhancing the associated gains per participant.

Regarding self-management, the collaboration of the participants in the Grid should be regulated by means of market mechanisms that provide them with the right incentives to both contribute to the Grid the resources promised and not to free-ride those of the others. For example, a global agreement can prescribe that all contribute as necessary. Similarly to peer-to-peer systems such agreements can be based either on rules prescribing a fixed minimum contribution for all participants or on rules regulating the consumption levels of each participant (quantitatively or qualitatively) in accordance/relation with his contribution over time. These rules should be complemented by accounting functionality that certifies conformance with them. Also, an internal market mechanism, based on SLA and monetary prices for these SLAs can also be employed as an effective approach for self-management, particularly in cases where the level of contribution of the various participants is not symmetric, and a global agreement is hard to be reached. These ideas apply to the case where the Grid is formed in order to serve the participants' own needs, including the case of a single organization with multiple departments (see item 1). If the participants also serve third parties, then the relations between the former and the latter should also be managed by means of market mechanisms.

3.3 Offering Utility Computing Services

This amounts to offering applications (software) and computing services (hardware) on a pay-per-use basis rather than by means of licensing or long term static agreements (leasing, etc.). In this model, applications are sold as components according to the SOA architectural concepts; customers can design their full solution by combining components from different providers and run these on their own premises or again using some Application Service Provider (ASP) computing

services. Essentially, this application level Grid allows for a new version of the application based on components to be accessed by the customers. This version is more affordable to infrequent users of the application, who now have a benefit compared to investing on the corresponding software license and/or computational infrastructure. Therefore, both these users and the service provider gain, since this version increases the demand for the service by making it affordable at lower costs. At the lower layers an ASP may benefit from Grid computing services using his own infrastructure complemented with utility computing services offered form third parties. The issues discussed in the previous items regarding high performance, economies of scale and scope etc. are still applicable here.

Nevertheless, other interesting economic issues arise too in the present case. In particular, we now have a new market (that of the pay-per use application), in which: a) the proper SLAs should be offered to customers, and b) resources should be self-managed and the revenue should be properly distributed to the players involved, while c) this market also has significant impact on other markets!

In case where this provider is a single organization, the self-management of its resources is attained through its incentives for optimizing its profits obtained from the market; for example, the predictions for market demand and the revenues foreseen accordingly can serve as an input of a capacity expansion policy. In case where the Grid provider is a virtual organization (or a single one yet with multiple participating departments), then additional self-management mechanisms are needed in order to pass the revenues to the various participants according to their level of contribution.

As already mentioned, the new market created in the present case may have a significant impact to other markets too. In particular, a Small and Medium Enterprise (SME) that cannot afford investing on a license or on infrastructure obtains new capabilities by outsourcing its missing application to the Grid provider on a pay-per-view basis. Thus, such an SME can now serve as a provider in another market, in which this application is a necessary capability for each provider. Therefore, the Grid version of the application leads to a reduction of the barriers of entry in the other market, which is now more competitive. This in turn may have a positive effect to the Grid provider itself, since the SMEs penetration in this new market generates additional demand for the Grid application. If beneficial for the Grid to expand, which is particularly the case if economies of scale and scope apply, then the customer SMEs will benefit even more by the expansion of Grid. Network externalities also apply here.

A summary of how the aforementioned issues impact the Grid adoption decision process is presented in the next table:

Table 1. The impact of the economic issues in the Grid adoption decision process
(1: Strong Influence, 2: Medium Influence, 3: Weak influence)

Categories/ Adoption Decision Influence	Econ. of Scale	Econ. of Scope	Network External ities	Self-manag ement	New markets	Impact to other markets	Free-riding	Info. Asym metry	Perform. Different iation
Optimisation of processing power	**1**	2	3	**1**	2	2	3	3	**1**
Sharing of complementary resources	**1**	**1**	**1**	**1**	2	2	2	2	2
Utility Computing	**1**	2	2	**1**	**1**	**1**	2	3	2

4 Analysing the Economic Issues Associated with Grid Business Scenarios

In the previous section we have briefly identified a number of economic issues that should be taken into account for Grid technology adoption in the business context. Consequently, these identified issues are listed below with a brief explanation of their meaning, their relevance in the context of Grid business scenarios and their significance. Following this analysis in subsequent sections we aim to evaluate them and further discuss their impact in terms of real-life Grid Business scenarios in Section 5. As will be seen there, these issues together with the objectives determine the decision of whether to adopt Grid or not; see Figure 1.

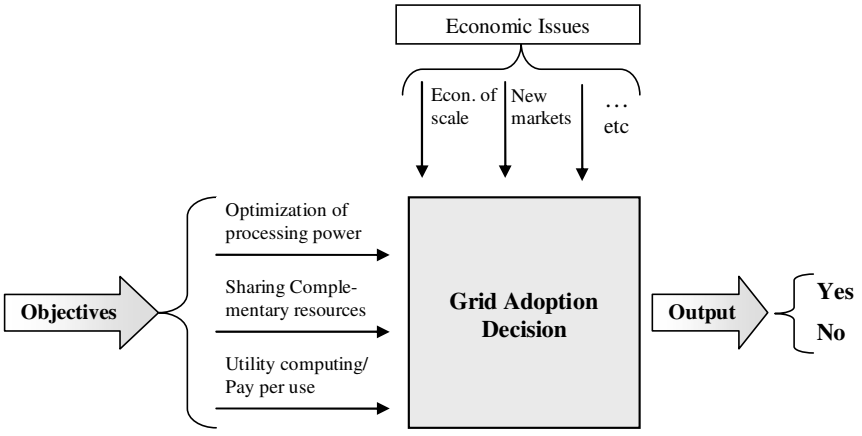


Fig. 1. Grid adoption decision process

Economies of scale and scope (complementarities)

As a definition it can be said that there are economies of scale in production if the cost per unit of production declines with the number of units produced. (Thus, "economies of scale" is a descriptive, quantitative term). Due to economies of scale, larger companies have greater access to markets in terms of selecting media to access those markets, and can operate with larger geographic reach whereas for traditional companies, size does have its limits, where additional size actually increases costs to companies (impacts communications costs etc., diminishing returns).

Economies of scope are conceptually similar to economies of scale. Whereas economies of scale apply to efficiencies associated with increasing the scale of production, economies of scope refer to efficiencies associated with broadening the scope of the service(s) offered, of marketing and distribution thereof etc. That is, while economies of scale refer primarily to supply-side changes (such as level of production), economies of scope also refer to demand-side changes (such as marketing and distribution).

The economic concepts of "economies of scope" and "economies of scale" similarly apply to the Grid market, where the integration of Grid technologies from a value chain actor and the consequent infrastructure and application improvements (e.g. in terms of performance) can lead to a production scale of the company's end-user products. Furthermore, sharing complementary resources among organizations could lead to the specification and the market entrance of new market products, thus to a realization of the "economies of scope" theory.

Network externalities

Network externalities are the effects on a user of a product or service of others using the same or compatible products or services. Positive network externalities exist if the benefits are an increasing function of the number of other users. Negative network externalities exist if the benefits are a decreasing function of the number of other users. For example a positive network externality arises in telephony, when the network expands; thus, each new user has more opportunities to communicate with others, and thus may be the amount that he is willing to pay for subscribing to the network depends on who or how many other parties are connected to it. Such an externality also applies to Internet, together with a negative externality that the more users the higher the congestion.

In a similar fashion, network externalities strongly apply to the Grid case, where for example an organization wishes to participate in a Virtual Organisation (VO) structure whose participants share complementary resources and the final outcome and thus "Grid" benefit for the new member strongly depends proportionally to the amount of resources available at that time to be shared by the other VO participants, i.e., the number of total members.

Self-management issues

Grid environments usually depict strong-collaboration principles especially where many players are involved e.g. different departments in an intra-organizational Grid structure or a VO. These players have a great deal of control on the Grid infrastructure and any change or management decision will produce an important effect for all of them. For this reason self-management of the Grid infrastructures and

services should apply in terms of how resources will be shared and on what charge so that the participant's incentives remain sound and solid. For example, an internal market mechanism such as a pricing unit ("Grid dollar") should essentially be defined as for the Grid resources to be shared according to well-defined principles and priorities.

New markets

By this criterion we refer to the possibility of the creation of new markets due to the use of Grid technology in existing or in new products, not foreseen before. For example, a company that was selling a software product or service to a customer based on specific commercial licenses (e.g. per machine installation), now can provide another version of the same service over a Grid infrastructure, without the customer having to install the software in his workstation, on a pay-per-use basis where the customer will pay for the times he uses the service only and depending on his requirements such as the QoS needed, the availability of the service, the completion time. The provider will make available different version of his service to accommodate the different requirements of his clients.

Furthermore, and due to the realization and wider adoption of the Grid technology a service provider could offer his product as a number of different stand-alone services (components) which the end-user can utilize together with services from other providers towards a new highly-customized and personalized scalable product or service. All players in this scenario take advantage of the new market foreseen by the realization of Service Oriented Architectures (SOA).

New entry and impact to other markets

As already mentioned, the new market created by the Grid adoption may have a significant impact to other markets too. The fact that applications are now offered on a pay-per-use basis provides new capabilities to SMEs, which can serve as providers to other markets, the barriers of entry to which are thus lower. Indeed, the SME can now develop applications and offer services over virtualised Grid environments (with fewer components, less actual development time and expensive infrastructure owned) and use the computing power of a Grid utility provider in order to offer them to a new market (not achieved before) thus directly impacting and increasing competition of this already established market.

Free-riding

In economics, collective bargaining, and political science, free riders are actors who consume more than their fair share of a resource, or shoulder less than a fair share of the costs of its production. The free rider problem is the question of how to prevent free riding from taking place, or at least limit its negative effects. Because the notion of 'fairness' is controversial, free riding is usually only considered to be an economic "problem" when it leads to the non-production or under-production of a public good or when it leads to the excessive use of a common property resource.

The problem and effects of free-riding are really evident in the context of a Grid virtual organization where resources are shared among and for the common benefit of their participants. A free-rider highly consuming participant limits the common benefit and participates on the expense of other participants. Hence, it is really

imperative for internal agreements e.g. SLAs to be implemented among VO-forming participants, the right incentives to be given to prevent free-riding, and penalties to be applied in cases where this is detected.

Information asymmetry, risk and unpredictability-related issues
Information asymmetry arises when one party to a transaction has more or better information than the other party. Typically it is the seller that knows more about the product than the buyer, however, it is possible for the reverse to be true: for the buyer to know more than the seller. Information asymmetry leads to market inefficiency, since not all the market participants do have access to the information they need for their decision-making processes.

In the context of Grid, information asymmetry and issues related to risk and unpredictability arise when participants of a Grid environment (either inter- intra-organizational) have incomplete information about the incentives and repudiation of other participants, such as VO members or internal company departments. This has a negative effect on their willingness to participate in the formation of Grid as well as on their reluctance in sharing their resources and data over the infrastructure. In all cases there is an associated risk and unpredictability of other partners' future behaviour and the origin of their incentives. Apart from the impact of information asymmetry on the sharing of resources by the Grid participants, security issues also impact their willingness to share data especially when sensitive information is to be distributed. This risk also applies to clients. Finally there is an always evident risk of adopting and investing on a new technology especially if this hasn't been fully adopted or if it is based on proprietary implementations.

Performance differentiation and QoS
The objective of performance improvement for application and services constitutes one of the foremost reasons for a company or organization in adopting/moving towards the Grid technology. Thus, it becomes apparent in such cases that the amount of money someone is willing to pay for a service provided over Grid or for such an implementation is strongly dependent on the magnitude of the advantage that this will offer to him in the market. The requirements from the clients/end-users may differ in terms of QoS parameters such as the time of completion, the availability and in this sense it is required to have different and adaptable (but secured by SLA-type agreements) level of services offered by the provider.

5 A Case Study: Analysing and Evaluating The BEinGRID Business Scenarios in Terms of the Identified Economic Issues

Following the identification of the main drives for Grid adoption and having elaborated on the economic issues around them, our next step is to classify the large number of possible Grid Business Scenarios in specific categories to enable us to discuss them further and investigate their impact in real-life scenarios.

In order to accommodate for business examples from different industries we have chosen to analyse the 18 business scenarios from the BEinGRID project (called Business Experiments –BEs in the context of the project). A high-level description of

these business cases can be found in [3]. The reasons behind our selection were the following:

- The BEinGRID business cases constitute real-life scenarios in the respect that are implemented by companies that their intention is to enter the Grid market immediately upon the successful completion of the project. Most of these companies did not have any previous experience with Grid Technologies and are currently in the phase of considering Grid adoption by evaluating all the relevant factors both business and technology oriented with emphasis on the former.
- The scenarios cover a range of industries from automotive and film industry to financial and ship building ones and including companies from the whole Grid services provisioning value chain: resource providers, integrators, service providers, end-users etc.
- As members of the BEinGRID consortium we had access to detailed (and sensitive) economic information such as their business models and business plans something that would not be available to us in any other case.

5.1 Classification of Grid Business Scenarios

Firstly, to ease the process of our analysis, the Business Scenarios were classified in three distinct categories, corresponding to the main economic objectives presented and discussed in the previous section. These categories are the following:

- *Category 1: "Grid Business Scenarios with a clear performance-associated benefit"*. This category of scenarios represents those cases that their implementations primarily aim at addressing one of the following limitations: a) additional CPU power needed for executing a demanding application (typical HPC scenario) b) huge amount of data storage/memory is required c) access to heterogeneous, geographically distributed data resources is required.
- *Category 2: "Grid Business Scenarios with a highly collaborative benefit"* i.e. benefit arising from sharing complementary resources among participating organizations. In this case the resulting benefit from Grid adoption comes from sharing data, power and resources utilized for a common scope. Typical examples of this category are intra-organisational Grids and Virtual Organisations and the expected economic benefit in this case could be shared among all participants in contrast to the first category where the main economic benefit is anticipated from the end-user where the service or application will be provided. Also, the services of this category cannot be provided by a single provider since data or other resources are necessary to be obtained from other providers.
- *Category 3: "Grid Business Scenarios exploiting the component-based software paradigm"*. This category comprises those business scenarios involving a service provider that offers applications on a pay-per-use basis rather than by means of licensing or long term static agreements and thus exploiting to the most the concepts of the next generation Service Oriented Architectures (SOA).

The classification of each of the BE's was based on analysing the technical context, business motivation and detailed work planned for the BE, as this was described in the relevant BEinGRID technical documents. In cases where a BE belonged to more than one categories, the decision was based on the prioritisation of the BE objectives as this was presented in the internal BE description and in some cases based on the feedback provided by us after contacting and interviewing the BE leading partner.

Our preliminary analysis with regard to the business scenarios and by examining their initial business plans provided to us in the context of the project, indicated that approximately 70% of the cases belonged to Category 1, 25% in Category 2 and only 5% in Category 1.

5.2 Impact of the Economic Issues in the Business Scenarios

Following the identification of the most important economic issues applicable to the Grid computing adoption in Section 4, the business scenarios were analysed in terms of these issues in order to investigate their relevance to the specific cases, the extent that these apply and therefore the importance that should be given to those by the partners involved in these experiments.

The BEs were evaluated using 3 different grades based on the applicability of each economic issue. The three grades were the following:

*Grade A – Strong Impact***:** Economic issues of this kind exist in this business case; their impact is very strong and should be carefully addressed

*Grade B – Average Impact***:** Economic issues of this kind may exist depending on the scenario configuration, or may exist in the future, their impact is and therefore should be analysed.

Grade C – Weak Impact: Economic issues of this kind do not exist or exist but their impact is considered weak and thus it is not vital to be considered at this point

Inputs for our evaluation were provided by the partners of the business experiments in terms of their business models and plans, technical descriptions of their scenarios and by personal interviews. The results of the evaluation for the experiments are presented in a tabled form in Appendix A.

5.3 Discussion on the Impact of Specific Economic Issues in the Business Scenarios

Our evaluation of the economic issues identified earlier with respect to the specific business scenarios resulted in a number of observations per economic issue examined. Due to space constraints only two of them are listed below as examples.

Network externalities

Network externalities are very evident in many of the experiments that involve the forming of a virtual organization to serve a common scope such as the execution of a complex simulation. The gained benefit for each organization is proportional to the number of organizations participating and offering their resources for the common

purpose. For example in "BE02: Business workflow decision making" in order for the risk simulations for the film production industry to be as comprehensive and sound as possible, information must be collected from many of the involved actors: film editors, special effects producers, animators etc – the more obviously the better. If the information is limited then the benefit for the end-user, i.e. the quality of risk-related results given to the producer, becomes questionable, thus decreasing the willingness of the producer in participating in such a virtual organization. The same characteristics can be found in BE10: Collaborative environment in the supply chain management where the number of participants increases the total benefit and vice versa, thus influencing the amount a potential customer is willing to pay for the same service. These observations are in line with our Category 2: "Grid Business Scenarios with a high collaboration benefit" economic characteristics discussed previously.

Information asymmetry, risk and unpredictability-related issues

As discussed in the previous section, information asymmetry and issues related to risk and unpredictability arise when participants of a Grid environment possess incomplete information about the incentives and repudiation of other participants, such as VO members or internal company departments. This has a negative effect in their willingness to participate in the Grid environment and in their reluctance for sharing their resources and data over the infrastructure. In these cases there is also an associated risk and unpredictability of the new partners' behaviour. These issues are more evident in the studied Grid scenarios where the Grid participants are not well-known before and depending on their numbers i.e. in the more "open/loose to participation" cases of Grid structures. On the other hand, more "closed" type of Grids, such the virtual organisations formed by a company's departments (enterprise Grids), are obviously less susceptible to such issues. Examples of the former are "BE10: Collaborative environment in the supply chain management" and "BE14: New product and process development" whereas of the later is "BE12: Sales management system".

For example, in BE14 let's consider a small firm that intends to run a complex CAD simulation for a potential new product. They have tried to run this simulation on their few workstations but couldn't complete it due to the insufficient power available from their machines. Using Grid technology i.e. "renting" infrastructure from a provider seems as an attractive option to them instead of buying new PCs or a new better CAD tool. However, their lack of expertise in computing and the fact that this is a new product does not enable them to estimate exactly the amount of CPU power and memory that will be needed from their CAD tool in order to perform these simulations. On the other hand, the computer experts from the Grid provider side, having used CAD tools extensively in the past and having rented their infrastructure to other companies for the same purpose in the past are in a better estimate the power needed for their simulation. If this information is not disclosed to the buyer (the small firm) could create a situation where they will pay to utilise more resources (to be on the safe side) than those actually needed for their product thus causing a market inefficiency.

6 Conclusion and Further Work

Grid technology has the potential to revolutionise the way services are distributed and executed over heterogeneous dispersed infrastructures in the future. Lessons learnt from recent past have taught us that technological maturity stand-alone cannot drive a new technology forward. Business and economical drivers should be considered as equally important. Along that direction, in this paper we have tried to identify and analyse a number of dominant economic issues that could act as both acceptance drivers as well as impediments and therefore should taken into account by industrial actors considering the adoption of the Grid for their business. These issues include the associated economies of scale/scope, information asymmetry, self-management issues, network externalities, free-riding, impact to new markets etc. We examined these in the context of a case study with real-life scenarios. Furthermore, we evaluated them in terms of their impact/influence in the decision process of whether a company should adopt the grid or not in the scenarios under consideration. Further work and analysis will include specific proposals on tackling these issues to be applied in an array of different industries. Finally, further work will include the definition of a decision model and associated methodology to be utilised by both Grid experts and business people for deciding towards the Grid adoption, based on the factors presented in Section 4.

Acknowledgements

The authors wish to thank T. Papaioannou and S. Routzounis for their contribution to the material of Section 2.

References

1. Forge S., Blackman C,.: Commercial Exploitation of Grid Technologies and Services, Drivers and Bariers, Business Models and Impacts of Using Free and open Source Licensing Schemes", SCF Accosiates for DG Information Society and Media (2006)
2. http://cordis.europa.eu/ist/grids/index.html
3. http://www.beinGrid.eu
4. http://cordis.europa.eu/fp6/
5. Foster I., Kesselman C., Nick J.M., Tuecke S.: The Physiology of the Grid. An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum (2002)
6. http://www.globus.org/ogsa/
7. G. Cheliotis, S. Miller, J. Woodward, and D. OH.: Questions for Getting Smarter on Creating a Grid Market Hub, Grid MarketPlace RoundTable. GECON'06, Singapore, May 2006
8. C. Kenyon, and G. Cheliotis: Grid Resource Commercialization, International Series in Operations Research and Management Science, Netherlands (2003), ISSU 64, 465-478
9. G. Cheliotis, and C. Kenyon: Autonomic Economics: Why Self-Managed e-Business Systems Will Talk Money, IEEE Conference on E-Commerce '03, (2003)
10. J. Gray: Distributed Computing Economics. Microsoft Research Technical Report: MSR-TR-2003-24 (also presented in Microsoft VC Summit 2004, Silicon Valey, April 2004), March 2003.
11. http:// www.Gridbus.org/

12. R. Buyya, D. Abramson, and S. Venugopal: The Grid economy, In Special Issue of the Proceedings of the IEEE on Grid Computing. IEEE Press, 2005.
13. J. Brunelle, P.Hurst, J.Huth, L.Kang, C.Ng, D.C.Parkes, M.Seltzer, J.Shank S.Youssef: Egg: an extensible and economics-inspired Open Grid computing platform, GECON 2006.
14. http://mnetproject.org/
15. Chuliang Weng, Minglu Li, Xinda Lu,Qianni Deng, Economic Based Resource Management Framework. CCGrid'05, 2005
16. J. K. MacKie-Mason, A. Osepayshvili, D. M. Reeves, and M. P. Wellman: Price Prediction Strategies for Market-Based Scheduling. In Proc. of ICAPS' 2004

Appendix A: Evaluation of the Impact of the Economic Issues in the BEs

Application	Economies of Scale	Economies of Scope	Externalities	Self-management	New markets	Impact to other markets	Free-riding	Info. Asymmetry	Perform. Differentiation
Computational Fluid Dynamics & Computer Aid Design	A	B	C	A	A	B	A	B	B
Business workflow decision making	A	C	A	A	A	B	B	B	A
Visualization & virtual reality	A	B	C	A	A	B	B	C	B
Financial Portfolio Management	A	A	C	A	A	A	B	A	A
Retail Management	A	C	C	A	B	B	A	B	A
Groundwater modelling	A	C	C	A	A	C	B	A	B
Earth Observation	A	A	B	A	A	B	A	B	B
Engineering and business processes in metal forming	A	A	A	A	A	B	A	B	A
Distributed online gaming	B	A	A	B	A	A	A	B	A
Collaborative environment in the supply chain management	A	B	A	A	B	B	A	A	B
Risk management	A	C	B	A	A	B	B	A	A
Sales management system	A	C	B	A	B	C	B	C	A
Textile Grid portal	A	A	A	A	A	A	A	A	B
New product & process development	A	B	B	A	A	B	A	A	A
Virtual engineering workplace for financial e-services	A	A	B	A	A	B	A	B	B
Ship building	A	C	C	A	B	B	A	B	A
Logistics & Distribution	A	C	B	A	A	C	B	C	A
Seismic imaging & reservoir simulation	B	A	A	A	A	B	A	A	B

Journal of Universal Computer Science, vol. 14, no. 5 (2008), 766-785
submitted: 1/11/07, accepted: 28/2/08, appeared: 1/3/08 © J.UCS

Dynamic Bandwidth Pricing: Provision Cost, Market Size, Effective Bandwidths and Price Games

Sergios Soursos, Costas Courcoubetis
(Athens University of Economics and Business, Greece
{sns, courcou}@aueb.gr)

Richard Weber
(University of Cambridge, UK
rrw1@statslab.cam.ac.uk)

Abstract: Nowadays, in the markets of broadband access services, traditional contracts are of "static" type. Customers buy the right to use a specific amount of resources for a specific period of time. On the other hand, modern services and applications render the demand for bandwidth highly variable and bursty. New types of contracts emerge ("dynamic contracts") which allow customers to dynamically adjust their bandwidth demand. In such an environment, we study the case of a price competition situation between two providers of static and dynamic contracts. We investigate the resulting reaction curves, search for the existence of an equilibrium point and examine if and how the market is segmented between the two providers. Our first model considers simple, constant provision costs. We then extend the model to include costs that depend on the multiplexing capabilities that the contracts offer to the providers, taking into consideration the size of the market. We base our analysis on the theory of effective bandwidths and investigate the new conditions that allow the provider of dynamic contracts to enter the market.

Key Words: network economics, contracts, pricing, bandwidth on demand, effective bandwidths, statistical multiplexing, reaction curves, competition, provision cost.
Category: C.2.0, G.1.6, J.4

1 Introduction

In the bandwidth markets, contracts between providers and customers have traditionally been of static nature. Both parties agree on the terms of a long-term contract, typically of one year, and comply to their contractual obligations throughout the defined time period. Such *static contracts* are widely used in the broadband access markets and for the formation of VPNs. The customers of the aforementioned services vary from small home users to large corporate users. Such contracts have the form of "buy X amount of bandwidth at a total price of $Y, for one year" or, a more generic form of "buy any amount of bandwidth at a unit price of $Y, for one year".

On the other hand, the nature of demand for such services cannot be characterized as 'static'. Especially for large customers (companies, organizations, etc), the demand is highly variable, depending on the type of applications/services that run above these lower-level (communication) services. Furthermore, new

technologies allow for dynamic allocation of network resources, based on the customer's current demand for bandwidth [Rabbat and Hamada 2006]. This change of paradigm renders *dynamic contracts* more suitable. A dynamic contract is also valid for a long-term period (e.g. for one year), but offers the customer the ability to buy bandwidth on shorter timescales (e.g. per week), according to his current needs and at a price, published by the provider, that remains constant throughout the long-term period, e.g. "buy any amount of bandwidth each week at a unit price of $Z, for one year".

The above contracts do not apply only to the case of a bandwidth market. Consider, for example, the case of a commercial Grid platform, a Grid marketplace, where providers and customers meet to buy and sell Grid resources. A static or dynamic contract in such a context would involve the purchase, or better leasing, of a number of virtual machines (VMs) for a specific time period. In such a market, a broker could offer longer term contracts, guaranteeing a fixed price and offering two alternative ways of acquiring resources: a fixed amount of VMs for a long period of time (static contract) or a more flexible contract for a variable amount of resources (dynamic contract) over a long period.

The basic properties of static and dynamic contracts in a monopoly environment have have been studied in our previous work [Courcoubetis et al. 2006]. In this paper, we examine the properties of such contracts under a price competition setup. Assume that there are only two providers, one that offers static contracts (*static provider*) and one offering dynamic contracts (*dynamic provider*). The providers must choose the unit price to publish in order to maximize their profits. Each provider takes into consideration the price that his competitor has published, in order to publish his own, profit-maximizing, unit price. Hence, a price competition situation emerges, where each provider is able to calculate a 'response' for every possible action his competitor takes, assuming the complete information case.

We also assume that both types of contract have the same duration. Thus, at the beginning of the long-term period, the customer must choose between a static and a dynamic contract, taking into consideration the unit prices that both contracts publish, as well as his (expected) requirements in bandwidth, in order to maximize his net benefit.

The most important factor that affects the provider's profit maximization problem is the provision cost. This cost defines what the unit price should be in order for the provider to be able to cover his costs and make some profits. We consider the case that both providers buy wholesale capacity (or dimension their networks) at the beginning of the long-term period, prior to fulfilling their contractual agreements. The provider of static contracts needs only to procure the exact amount of capacity he has sold in his contracts, since his customers cannot change their demand for the duration of the contract. On the contrary,

the provider of dynamic contracts has to procure more capacity than the expected average requested in each short-term period, so as to be able to fulfill his contractual obligations. So, it is obvious that the provision cost of a unit of bandwidth for a static provider is less than the unit cost of the dynamic provider.

In our work, we consider two cases as far as the provision cost is concerned. Initially, we provide a summary of the case where the provision costs are constant and independent of the market size. We then consider the case where the provision cost of the dynamic provider depends on the market size and the multiplexing opportunities that dynamic contracts offer. According to the effective bandwidths theory and due to the variable nature of demand under a dynamic contract, the amount of bandwidth reserved per customer in the bandwidth inventory tends to the average bandwidth consumption of the contract as the size of the market increases, i.e. as the number of dynamic contracts served increases. Hence, our purpose is to capture the effects of statistical multiplexing in the cost structure of the dynamic provider.

The main results of this paper involve the operating point of the market. In other words, for both cases of constant and multiplexing-dependent costs, we examine if and when there exists an equilibrium in the price competition game and how the market is segmented at this point. Especially for the more interesting case of the multiplexing-dependent provision costs, we examine what is the necessary size of the market for a dynamic provider to enter and become active as the provision costs increase, what are the profits of the providers as the market size increases and how the amount of the bandwidth reserved per customer changes with the market population.

Competition between providers has also been studied in [Mason 2000], where it is shown that flat rate pricing is more efficient than usage-based prices in equilibrium, under certain conditions. Note that our pricing model is related to the above model since providers compete by changing their pricing schemes for usage: in our model the customers decide on the size of their contracts both in the static and in the dynamic case. But it is also remotely related to usage based pricing, since customers are not charged for their actual usage. The form of contracts described earlier can be more of a flat rate charging, although the resource consumption by the costumers is not unrestricted. Competition between providers has been studied in relation with service compatibility as well. [Foros and Hansen 2001] model a two-stage game in a duopoly setup, where the providers first choose the level of service compatibility and then decide on prices. They show that increased compatibility reduces competitive pressure due to network externalities. [Gibbens et al. 2000] have shown that 'multiproduct' outcomes of a competition between providers always offer lower profits to them, as opposed to the case of a single service class. In our context, providers offer compatible services, i.e. a single service class, but compete in order to gain

as many customers as possible, by offering different pricing schemes for usage. In [Davies et al. 2004], the growing necessity for capacity planning and pricing in packet switched networks is discussed. The relation between statistical multiplexing and effective bandwidths is explained in [Courcoubetis and Weber 2003].

The remainder of this paper is organized as follows. Section 2 presents the main assumptions and defines the demand model. In Section 3, a price competition game is analyzed, where the players (the providers of static and dynamic contracts) have constant provision costs. We define the profit maximization problems and provide the equations for the reaction curves. In Section 4, the game setup is altered in order for the provision costs of the dynamic provider to depend on multiplexing. Effective bandwidths are explained and introduced to our model in Section 4.1. Section 5 shows the main numerical results from the analysis of the model. In Section 5.1 the reaction curves are plotted and the equilibrium points are examined, for various market sizes and unit costs. In Section 5.2, the profits are plotted while in Section 5.3 the allocated bandwidth per customer is examined. Section 5.4 discusses about the market segmentation. Finally, we conclude in Section 6 and discuss issues for further study.

2 Demand model and assumptions

Assume that time is slotted, starting from slot 0 up to slot m. The customer decides what type of contract to make just before slot 0. In the case of static contracts, the amount of the bandwidth purchased will be chosen at that time and will remain constant for the remaining m slots. In the case of dynamic contracts, the customer will choose the bandwidth to be purchased at the beginning of each slot i.

To capture the interesting phenomena in this market, we use a very simple form of linear utility function, with slope -1, which varies randomly between slots. The customer's utility for consuming x units of bandwidth in slot i is $u_k(x)$, where k is a parameter, presently unknown, but distributed a priori as a random variable with a known distribution function $F(k)$. To obtain the simple form of demand mentioned before, we assume that the utility function has the following form:

$$u_k(x) = \begin{cases} kx - \frac{1}{2}x^2 \ , & \text{if } x \leq k \\ \frac{1}{2}k^2 \qquad\quad , & \text{if } x \geq k \, . \end{cases} \tag{1}$$

This utility function is, by design, increasing and concave, as the first segment of (1) suggests. The second segment implies that utility reaches a maximum and then continues to stay there, even if the the consumption increases. If the second term did not exist, the utility would be decreasing, for $x \geq k$, which is not a desired property in our case.

We assume that, at the beginning of each slot, the value of k becomes known to the customer. If the customer faces a price of a and knows k then he will

choose x to maximize $E[mu_k(x) - ax] = mE[u_k(x) - \hat{a}x]$, where $\hat{a} = a/m$ is the static price scaled per slot. For simplifying notation, we use this scaled price as the price a of bandwidth under a static contract. Similarly, without loss of generality, we can assume $m = 1$, since the form of the optimization problem remains the same. This gives the demand function $x_k(a) = \max\{k - a, 0\}$ (see Fig. 1).

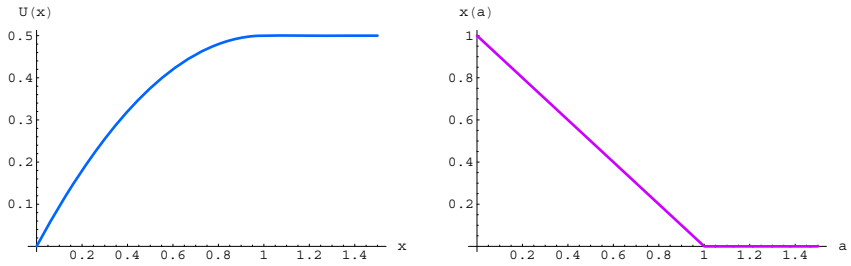


Figure 1: The utility function and resulting demand function when $k = 1$.

In order to illustrate the various issues to be studied and capture the varying bandwidth needs of a customer, we examine a simpler case where k takes discrete values and, more specifically, it is a random variable that follows a two-point distribution. That is, with probability $1 - p$ his utility will be $u_{k_1}(x)$ and with probability p the customer will have a utility of $u_{k_2}(x)$. We suppose $p \in [0, 1]$ is distributed across the population of n customers with a density function of $f(p)$.

The two-point distributed k approach is a simplification of customer's actual behavior regarding network access services. At the same time, it is more realistic than having a constant demand through time. As already mentioned, current applications and services have different requirements in bandwidth. As a result, customers have bursty demand, depending on the applications and services they use each time. In fact, our simplification models a situation where the customer has a minimum and a peak requirement for access bandwidth, without knowing a priori when such needs will occur.

To make things more simple for the analysis to follow, we assume that for any given customer it holds that $k = 0$ or $k = 1$, with probabilities $1 - p$ and p respectively. As already mentioned, the provider of static contracts sells at price a, but the contract must be made prior to the customer's knowledge of the exact allocation of high and low demand periods in the long-term period, i.e. the distribution of k. The dynamic provider sells at price b, with the flexibility that the customer needs not to make the purchase until the beginning of the next slot, when he knows whether his k will be equal to 0 or 1.

When making static contracts, the customer chooses x to maximize $pu_1(x) - ax$ and so optimally buys $\max\{1 - a/p, 0\}$. When making dynamic contracts, the customer chooses x to maximize $pu_1(x) - pbx$ and optimally buys $\max\{1 - b, 0\}$. Thus a customer strictly prefers a static contract rather than a dynamic one if and only if $a < pb$.

This simplification on the discrete values that k can take (zero or one), does not render the results unrealistic, as related to the more generic case of k_1 and k_2. In fact, in the more generic case, the customer will surely buy k_1 units of bandwidth and the actual decision is how much above the k_1 units will he buy (under either type of contract). Hence, the extra bandwidth will belong to $[0, k_2 - k_1]$. Our only simplification is that we use 1 as an upper bound of the difference.

3 Market equilibrium: Constant cost model

Considering the previous condition for the choice between the two types of contracts, we assume that each provider has a different but constant provision cost. More specifically, the static provider has a unit cost of c_1 and the dynamic provider a unit cost of c_2 which is constant and insensitive to the fact that the provider must provide a total amount of bandwidth that is fluctuating between slots. As explained in the introduction, we assume that it holds $c_2 > c_1$.

Hence, the average profits obtained by the two providers per slot are

$$\text{prof}_S(a,b) = \begin{cases} n(a-c_1)\int_{a/b}^{1}(1-a/p)f(p)\,dp\,, & \text{if } a < b \\ 0 \qquad\qquad\qquad\qquad\qquad\quad\;\; , & \text{if } a \geq b\,, \end{cases} \tag{2}$$

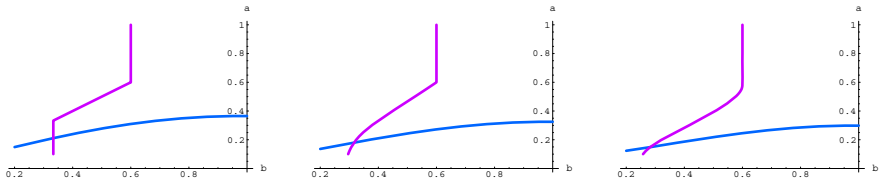
$$\text{prof}_D(a,b) = \begin{cases} n(b-c_2)\int_{0}^{a/b}p(1-b)f(p)\,dp\,, & \text{if } a < b \\ n(b-c_2)\int_{0}^{1}p(1-b)f(p)\,dp \quad , & \text{if } a \geq b\,, \end{cases} \tag{3}$$

where n is the number of customers in the market. Note that n does not affect the profit maximization problem, hence the result is the same even if there was even only one customer.

Equations (2) and (3) provide the payoffs in a Bertrand game of price competition [Binmore 1991]. From these, we can compute the reaction curves:

$$a(b) = \arg\max_a\{f_A(a,b)\}, \qquad b(a) = \arg\max_b\{f_B(a,b)\}\,.$$

To give some numerical examples, suppose $c_1 = 0.1$ and $c_2 = 0.2$. Figure 2(a) shows the reaction curves when p is uniformly distributed, i.e., $f(p) = 1$. Figure 2(b) shows these curves when $f(p) = 6p(1-p)$. Figure 2(c) shows these curves when the distribution is more concentrated around $p = 1/2$, with $f(p) = 630p^4(1-p)^4$. The point of intersection in each graph is a Nash equilibrium. In



(a) (b) (c)

Figure 2: The reaction curves $a(b)$ against b (in blue), and a against $b(a)$ (in violet), for different distributions of p.

the equilibrium point of Fig. 2(a), $a = 0.2105$, $b = 0.3333$ and the respective revenues are 0.0300 and 0.0177.

For the case of a uniformly distributed p, we can provide the function of the $b(a)$ curve and the maximization problem from which we obtain the function of the reaction curve $a(b)$. More specifically, having $f(p) = 1$ and taking into consideration (3), the provider of dynamic contracts expects that the average amount of bandwidth bought by each customer who prefers a dynamic contract is

$$x_D(a,b) = \begin{cases} \frac{1}{2}\frac{a^2}{b^2}(1-b)\,, & \text{if } a < b \\ \frac{1}{2}(1-b) \quad\;\; , & \text{if } a \geq b\,. \end{cases}$$

Hence, solving the revenue maximization problem for the provider of dynamic contracts

$$\underset{b}{\text{maximize}}(b-c_2)x_D(a,b) \quad \text{w.r.t. } c_2 \leq b\,,$$

we have that

$$b(a) = \begin{cases} \frac{2c_2}{1+c_2}\,, & \text{if } a < \frac{2c_2}{1+c_2} \\ \frac{1+c_2}{2}\,, & \text{if } a > \frac{1+c_2}{2}\,. \end{cases} \tag{4}$$

The only case that remains to be examined is what value b takes when $\frac{1+c_2}{2} \geq a \geq \frac{2c_2}{1+c_2}$. In this range, for every $b > a$ the profits increase as b decreases. When b becomes lower than a ($b < a$), the profit maximization problem changes and profits increase as b increases. Hence, the profit maximizing value of b, in the interval $[\frac{1+c_2}{2}, \frac{2c_2}{1+c_2}]$, is where $b = a$.

From (2) we get that the amount of bandwidth bought by each customer that prefers a static contract is

$$x_S(a,b) = \begin{cases} 1 - \frac{a}{b} + a\ln\frac{a}{b}\,, & \text{if } a < b \\ 0 \qquad\qquad\qquad\;\;, & \text{if } a \geq b\,. \end{cases}$$

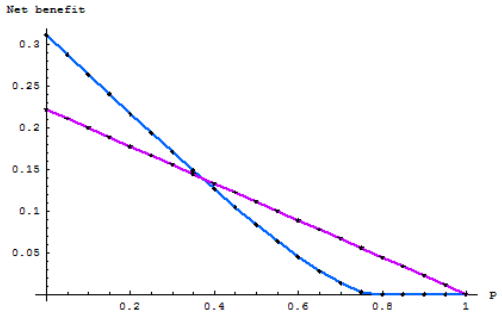


Figure 3: The customer's net benefit for various values of p, with $c_1 = 0.1$ and $c_2 = 0.2$. The violet line depicts the net benefits under a dynamic contract while the blue line shows the net benefits under a static contract. We observe that both providers gain a share of the market, with the dynamic provider acquiring the customers with a high value of p.

Hence, the provider of static contracts wants to maximize his profits, i.e. chooses a price $a(b)$ which solves

$$\underset{a}{\text{maximize}}(a - c_1)x_S(a, b) \quad \text{w.r.t. } c_1 \leq a\,.$$

The exact formula for $a(b)$ cannot be deduced by taking the partial derivative w.r.t a of the above maximizing function equal to zero, due to the complexity of the problem. But the numerical results reveal the form of the curve and the existence of an equilibrium point. As far as the way the market is segmented, Fig. 3 shows how the net benefits of a customer vary with his type p, under a static and a dynamic contract. Hence, the way the market is segmented is obvious: for low to medium values of p, the customers prefer the provider of static contracts over the provider of dynamic contracts. This preference is reversed for higher values of p. The fact that the market is segmented, opposes to the classic theory of Bertrand games where one provider (the one with the lower costs) attracts all the customers.

4 Market equilibrium: Multiplexing-dependent cost model

In the previous section, we have considered the case where the unit provision costs under static or dynamic contracts are constant. As already mentioned in the introduction, c_1 and c_2 may not be arbitrary. The size of the customer base and the provider's multiplexing capabilities affect the provision costs, especially for the provider of dynamic contracts.

It is reasonable to assume that under static contracts, the provider reserves the total amount of bandwidth purchased by the customers. Hence, it is relatively easy for a provider of such contracts to plan the capacity of his network, based on the number of his customers and their contracted capacity. On the other hand, under dynamic contracts, users express demand that varies with time. This results in a variable demand for the provider to serve over a long period. Hence, capacity planning is no longer obvious, since the dynamic provider must estimate how much bandwidth to reserve per dynamic contract. Reserving the maximum that a customer may request, may lead to a waste of bandwidth, since it is highly improbable that all customers will simultaneously request this maximum value. If he uses overbooking, then he reduces his provision cost but may incur a cost of not fulfilling his contractual commitments.

It follows that the amount of bandwidth the dynamic provider needs to reserve per contract is higher than the average consumed by the contract. This amount corresponds to the effective bandwidth of the bandwidth demand process generated by the contract over time. We discuss this in the next section.

Suppose now that both providers must define a bandwidth inventory for satisfying the needs of their contracts and that this inventory will be of some fixed size C, at a unit cost c, same for both providers. In the case of supporting static contracts, the size of the inventory will be equal to the total requested demand of the signed contracts. In the case of dynamic contracts, the amount reserved per customer will need to be larger than the average bandwidth that the provider will need to offer per slot to that customer.

It is hence reasonable to assume that in the case of dynamic contracts, the unit cost is directly affected by the number of customers a provider serves. That is, a higher number of customers will lead to lower provision costs since their demand can be better multiplexed by the inventory than in the case of fewer customers. Ideally one could say that, for infinite number of customers, the provision unit cost of such a provider is the same with the provision unit cost of a static contract, since he needs to reserve an amount close to the average amount of bandwidth per contract. The above statement is based on the fact that the more traffic flows are, the more efficient multiplexing becomes.

4.1 Effective bandwidth analysis

Our purpose is to include the relationship between costs and market size into the model of the price game between a static and a dynamic provider. In fact, our goal is to include the notion of effective bandwidth in the definition of the provision cost for the provider that offers dynamic contracts.

In order to do so, we denote with c the cost for providing one unit of bandwidth in the bandwidth inventory. Since the provider of static contracts needs to maintain an inventory equal to the sum of the bandwidth sold in the contracts,

the cost of a unit offered is exactly the same with the unit cost of the bandwidth inventory ($c_1 = c$).

For the provider of dynamic contracts, the total profit can be expressed as

$$\text{prof}_D(a,b) = \begin{cases} n\ b \int_0^{a/b} p(1-b)f(p)\,dp - c\ C(n,a,b)\ , & \text{if } a < b \\ n\ b \int_0^{1} p(1-b)f(p)\,dp - c\ C(n,a,b) & , \text{ if } a \geq b\,, \end{cases}$$

where $C(n,a,b)$ is the size of the inventory, as a function of the number of customers acquired and the unit prices published by both providers.

Our first task is to define the capacity needed by a provider of dynamic contracts in order to fulfill the requirements of the n customers that reside in the market. Note that not all n customers will choose a dynamic contract, but we will address this issue later on. We use the theory of effective bandwidths in order to express the required total capacity C as a function of n and the expected bandwidth requirements by each user, given a QoS target. In particular, our QoS target defines the probability that the provider of dynamic contracts will not be able to honor all his contractual obligations, which can occur if at a particular slot the sum of the demands of his customers exceeds C, the size of his inventory (i.e. the demand will overflow the inventory). We call this the Inventory Overflow Probability (IOP).

We can express the QoS target with the following inequality:

$$P[\sum_{i}^{n} x_i \geq C] \leq e^{-\epsilon}\,, \tag{5}$$

where $e^{-\epsilon}$ denotes the IOP and x_i the bandwidth requested by customer i at a random slot.

From the theory of Chernoff bounds, we have that for a sequence of i.i.d. random variables $x_1,\ldots,\ x_n$ it holds:

$$P\left[\sum_{i}^{n} x_i \geq C\right] \leq E\left[e^{\theta(\sum x_i - C)}\right]\,, \forall\theta\,. \tag{6}$$

The right-hand side of the above inequality can be re-written as follows:

$$E\left[e^{\theta(\sum x_i - C)}\right] = e^{-n\left(\theta C/n - \log E\left[e^{\theta x_i}\right]\right)}\,. \tag{7}$$

Hence, we obtain that

$$P\left[\sum_{i}^{n} x_i \geq C\right] \leq e^{-n\sup_\theta\left[\theta C/n - \log E\left[e^{\theta x_i}\right]\right]}\,. \tag{8}$$

Combining (5) and (8), the sufficient condition for the QoS target to hold is

$$e^{-n\sup_\theta\left[\theta C/n - \log E\left[e^{\theta x_i}\right]\right]} \leq e^{-\epsilon} \Rightarrow \sup_\theta\left[\theta C - n\log E\left[e^{\theta x_i}\right] - \epsilon\right] \geq 0$$

$$\Rightarrow \sup_\theta\left[C - \frac{n}{\theta}\log E\left[e^{\theta x_i}\right] - \frac{\epsilon}{\theta}\right] \geq 0 \Rightarrow C - \inf_\theta\left[\frac{n}{\theta}\log E\left[e^{\theta x_i}\right] + \frac{\epsilon}{\theta}\right] \geq 0\,. \tag{9}$$

The above inequality defines the minimum capacity that the provider should have available in his access network in order to guarantee a QoS target of $IOP = e^{-\epsilon}$ in his contracts with the customers.

What remains to be defined is how we obtain x_i from our customer demand model in order to calculate the $E\left[e^{\theta x_i}\right]$ factor, from which we calculate the total capacity needed by solving the minimization problem in (9).

We have seen that if a customer's p_i is greater than a/b, then the customer will choose a static contract. This knowledge can be used in the calculation of the effective bandwidth. Thus, instead of considering all the customers in the market (n), we only examine those that prefer dynamic contracts (n'). It holds that $n' = na/b$, on average.

The distribution of bandwidth that a customer of dynamic contract requests is now more simple to express: with probability p_i he buys $1 - b$ units of bandwidth, while with probability $1 - p_i$ he does not buy nothing at all, with p_i being uniform in $[0, a/b]$. Hence, $E\left[e^{\theta x_i} | p_i = p \leq a/b\right] = pe^{\theta(1-b)} + (1-p)$.

Thus, we have that

$$\begin{aligned} E\left[e^{\theta x_i}\right] &= E\left[E\left[e^{\theta x_i} | p_i = p \leq a/b\right]\right] = \tfrac{b}{a}\int_0^{a/b}\left(pe^{\theta(1-b)} + (1-p)\right)dp \\ &= \left(\tfrac{a^2}{2b^2}e^{\theta(1-b)} + \tfrac{a}{b} - \tfrac{a^2}{2b^2}\right)\tfrac{b}{a} = \tfrac{a}{2b}e^{\theta(1-b)} + 1 - \tfrac{a}{2b}\,. \end{aligned}$$

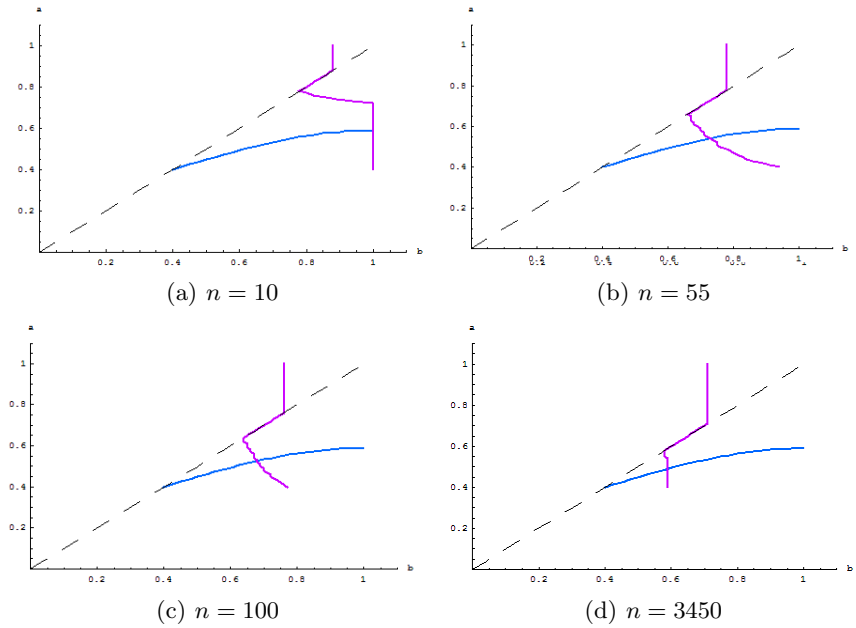
Note that the factor b/a in the integral is actually the probability density function of p_i since now p_i is uniformly distributed in $[0, a/b]$. From the above, we get that the required capacity for the inventory that a provider of dynamic contracts needs to purchase, so as not to violate the IOP target is given by the following formula (note that n is now replaced by na/b):

$$C \geq \inf_{\theta}\left[\frac{\epsilon}{\theta} + \frac{a}{b}\frac{n}{\theta}\log\left(\frac{a}{2b}e^{\theta(1-b)} + 1 - \frac{a}{2b}\right)\right]. \tag{10}$$

5 Market analysis

In this section we present some of the most representative results using the previous model to analyze the behavior of the market, the equilibrium point, the profits of both providers and the market segmentation. Our purpose is to compare the case of provision affected by the number of customers in the market (especially when n is small) with the earlier case where we assumed constant costs, which corresponds to $n = \infty$.

More precisely, the analysis setup involves a price competition game between a provider offering static contracts and a provider offering dynamic contracts. The system parameters are the cost c for the provision of one unit of bandwidth and the number n of customers in the market. We assume that IOP is fixed and equal to 0.01. Each provider calculates a price as the best response (in terms of revenue maximization) at every possible price his competitor may post.



(a) $n = 10$ (b) $n = 55$

(c) $n = 100$ (d) $n = 3450$

Figure 4: The reaction curves $a(b)$ (blue) and $b(a)$ (violet) for $c = 0.4$. We obtain that for small n the provider of dynamic contracts does not participate in the market.

5.1 Reaction curves and equilibrium points

First of all, we examine the evolution of the reaction curves as the number of customers increases. It will be very interesting to see what happens for a small number of customers were multiplexing is not so efficient and effective bandwidths are large. Obviously, only the $b(a)$ reaction curve will change, since there is no change in the way the provider of static contracts decides his best response a to a published price b by the provider of dynamic contracts.

Figure 4 shows the reaction curves of the two providers. The blue curve depicts the reaction of provider that offers static contracts. The red curve depicts the reaction of the provider that offers dynamic contracts. The unit cost is set to be $c = 0.4$. The size of the market varies from very small ($n = 10$), to medium ($n = 55$ and $n = 100$) and very large ($n = 3450$).

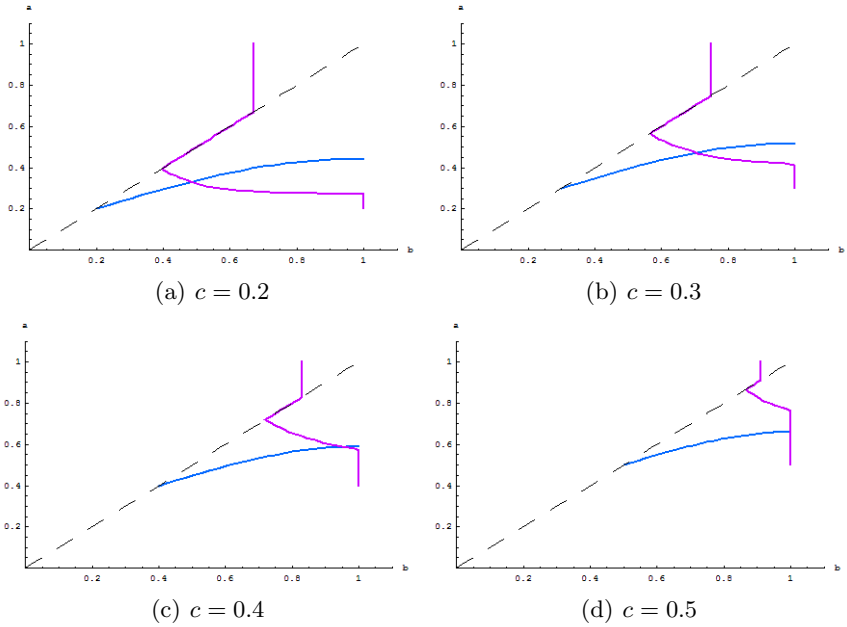
When the size of the market is very small, the provider of dynamic contracts cannot take full advantage of multiplexing. Thus, the effective bandwidth per customer is very high and it is not beneficial for the dynamic provider to participate in such a market. This explains why, for $n = 10$, we get that $b = 1$ at the equilibrium point, which means that no customer will prefer a dynamic contract

at such a high price. Recollect that the demand function for a single time slot is $1 - b$, leading to zero consumption when $b = 1$. As the size of the market increases, multiplexing becomes more efficient, provision costs decrease and the dynamic contracts become beneficial for some customers. Hence, at the equilibrium point it will hold that $b < 1$. Finally, when the market size becomes quite large, the respective reaction curves tend to become identical with the reaction curves of the initial model with fixed provision costs, where $n = \infty$ (compare with Fig. 2(a)). This happens due to the fact that the market is so large that the number of customers does not affect the provision cost any more, since their flows are ideally multiplexed and the effective bandwidth is equal to the average bandwidth a customer requests.

The new reaction curve $b(a)$ has three segments, similar to the reaction curve of Section 3. In the first segment, for very high values of a, both approaches provide the same, fixed value for b. Then, in the second segment, as a decreases, b also decreases and in fact it holds that $b = a$, for the same reason explained in Section 3, where we had that $n = \infty$. At the third segment, i.e. for lower values of a, b increases, since the provider of dynamic contracts has to cover his large provision costs and cannot continue decreasing b. This is also due to the fact that the probability for a customer to select a dynamic contract ($p < a/b$) is becoming lower. For a better understanding of how the profits under a dynamic contract change when b varies and a remains constant, consult Section 5.2.

As far as the existence of an equilibrium point is concerned, we have seen that as the size of the market changes, the equilibrium point gets closer to the equilibrium point calculated when having fixed unit costs (see Section 3). We have also explained what happens for small market sizes, where it is not beneficial for a provider who offers dynamic contracts to participate. We need to show whether the existence of an equilibrium depends on the value of c.

Figure 5 depicts how the equilibrium point moves as the provision unit cost increases, for a small market ($n = 20$). It is obvious that as the cost increases, all reaction curves are shifted upwards and the reaction curves of the dynamic provider are also shifted to the right. As a result, increasing unit costs result in higher prices and for very high costs, the provider of dynamic contracts has no benefit in participating in such a market. Simulations have also shown that for large-sized markets ($n > 2500$), a non-operational equilibrium point will be reached only for extremely high unit costs ($c > 0.9$). To illustrate this better, Fig. 6 shows the minimum market size n that makes it profitable for the dynamic provider to enter the market, as c increases. It is obvious that the minimum market size is growing exponentially, meaning that unit cost must acquire very high values so that even when the market is large market, it is not beneficial for the dynamic provider to be activated in such a market.



(a) $c = 0.2$ (b) $c = 0.3$

(c) $c = 0.4$ (d) $c = 0.5$

Figure 5: The reaction curves for the providers of static(blue) and dynamic(violet) providers, as c increases and $n = 20$. As costs increase the dynamic provider is discouraged from joining the market ($b \to 1$).

5.2 Provider's profits

An interesting property to examine is how the profits of the dynamic provider change as prices a and b change, in order to understand how the reaction curve $b(a)$ is formed. Our purpose is to investigate the properties of each segment of the $b(a)$ reaction curve so as to understand better its shape. As a reference, we will use Fig. 4(c), where $n = 100$ and $c = 0.4$.

Figure 7 shows how the profits change for fixed values of a, as b varies. The first obvious result is that while $b < a$, as b increases so do the profits of the dynamic provider. This forms the first segment of the profit curve, which is either linear or concave. When b becomes larger than a, we have the second segment of the profit curve, which is also linear or concave, depending again on the value of a. Hence, the profit maximizing point is either where the first and second segment meet, or strictly on the second segment.

For high values of a, the profit maximization point remains fixed (see Fig. 7(a) and 7(b)). This explains the first linear segment of the reaction curve $b(a)$. As seen in the figures, the maximizing point is where the two segments of the profit curve meet. As a continues to decrease (second segment of the reaction

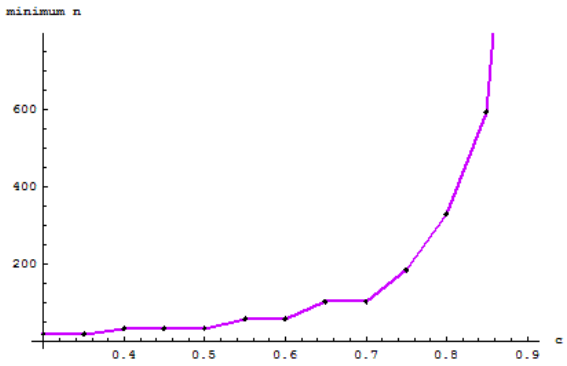


Figure 6: Minimum market size required for the provider of dynamic contracts to enter the market, as c increases. The exponential behavior of the minimum required market size is obvious.

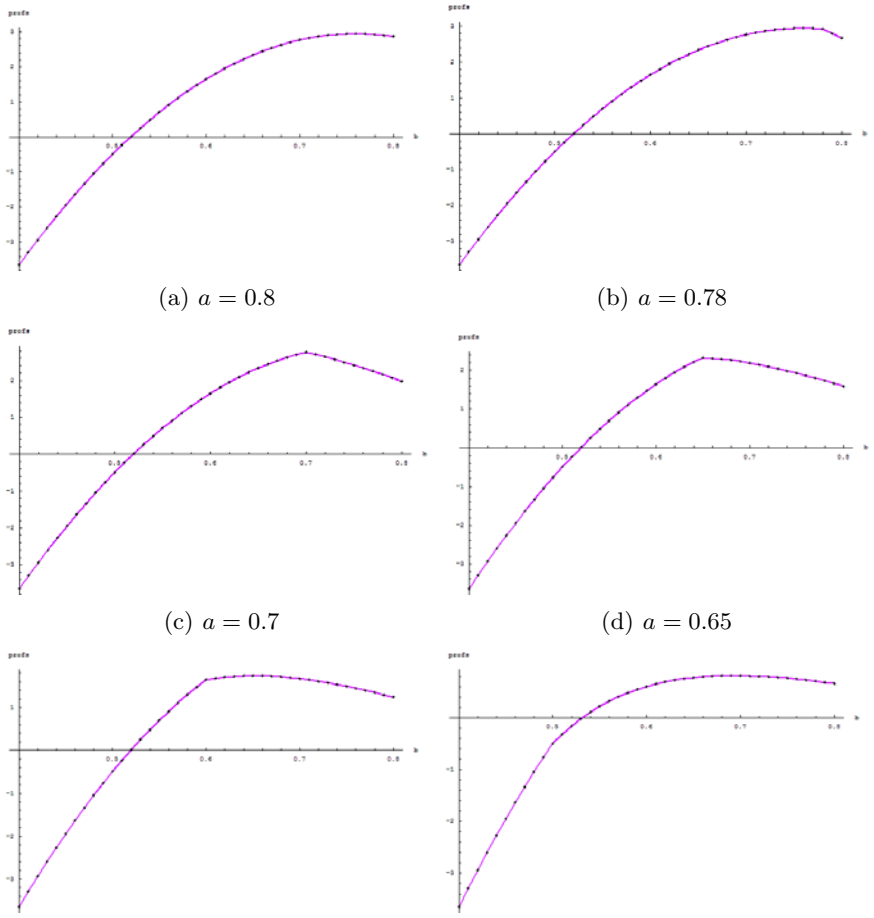
curve), the maximizing b becomes equal to a and starts decreasing (see Fig. 7(c) and 7(d)). In these figures, we observe that the maximizing point is still at the meeting point of the two segments of the profit curve, but this point moves to the left, as a decreases, resulting in a smaller linear segment and a more concave second segment. There is a point where the maximizing value of b becomes the local maximum of the second segment. Thus, the optimal b starts increasing as a decreases (third segment of the reaction curve) (see Fig. 7(e) and 7(f)).

It is also interesting to see how the profits of the static and dynamic contracts are related, especially for small markets. It is expected that the larger the market, the higher the profits of the dynamic provider will be. In Fig. 8, we depict the providers' profits for small-sized market. As expected, for very small markets, the provider of static contracts obtains higher profits since the dynamic provider either does not participate in the market, or his unit price b is large enough in order to cover his provision costs.

So far, we have illustrated the reaction curve of the dynamic provider when in a price competition environment. We have also seen, how the profit maximization problem affects these reaction curves. In the next section, we will show what happens with the bandwidth provision considerations for the provider's bandwidth inventory.

5.3 Bandwidth allocation comparison

As mentioned earlier, the per customer bandwidth provision is expected to be higher than the average bandwidth per user and lower than the peak bandwidth



(a) $a = 0.8$ (b) $a = 0.78$

(c) $a = 0.7$ (d) $a = 0.65$

(e) $a = 0.6$ (f) $a = 0.5$

Figure 7: The profits curve for fixed values of a and varying b ($n = 100$). The maximum in each curve defines how the reaction curve $b(a)$ is formed and explains its three linear segments.

that a customer may require. In this section, we show that the numerical analysis of our model is validated by the theory of effective bandwidths.

First, we examine the amount of bandwidth that the provider of dynamic contracts needs to reserve in his inventory for each customer, when using peak, average and effective bandwidth reservation techniques. In Fig. 9, we keep a and b fixed and we only vary the size n of the market.

From the above diagram, it is obvious that the effective bandwidth is always

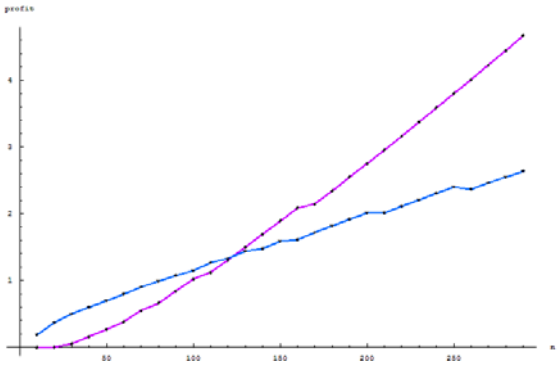


Figure 8: Total profits for the providers of static and dynamic contracts, as the market size increases. Violet line represents profits of the dynamic provider while blue line shows the profits of the static provider ($c = 0.4$). For small market sizes, static contracts are more profitable. The result is reversed for larger markets.

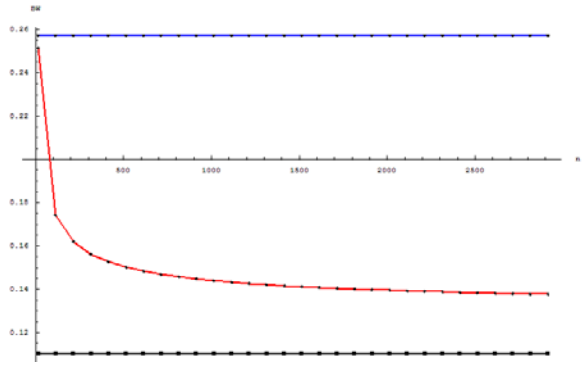
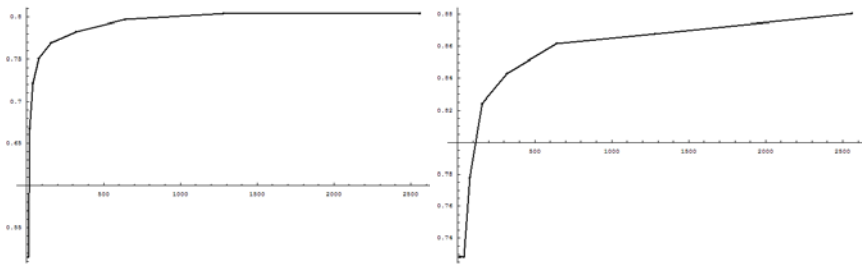


Figure 9: The per customer effective (red), average (black) and peak (blue) bandwidth needed to be reserved, for fixed prices and unit cost ($a = 0.6$, $b = 0.7$ and $c = 0.3$), as n increases. Observe that the effective bandwidth tends to become equal to the average, as n increases.

between the peak and average bandwidth, as expected. Furthermore, as the number of customers increases, the effective bandwidth converges to the average bandwidth. This result is in accordance with the theory of effective bandwidths. The variation of the unit cost affects only the actual bandwidth allocated and not the relative position of the curves.

5.4 Market Segmentation

One last issue, is how the market is segmented between the providers offering static and dynamic contracts. Recollect that, whether a customer chooses a dynamic or static contract, depends on his type, i.e. the value of p, and the unit prices of the market. So, if $p > a/b$ then the customer prefers a static contract. This preference is reversed when $p \leq a/b$. In Fig. 10, we plot the a/b value, for two unit costs ($c = 0.3$ and $c = 0.6$) as the market size changes.



(a) $c = 0.3$ (b) $c = 0.6$

Figure 10: The market segmentation threshold (a/b) for fixed values of c, as n increases. The portion of customers preferring dynamic contracts is above half, even for smaller market sizes.

We observe that a provider of dynamic contracts obtains a large portion of the market. This portion increase as the market size increases and becomes almost fixed for large markets. Even in a situation with low provision costs, such a provider obtains more than 70% of the market share. And this happens even in relatively small markets (e.g. for $n = 100$).

6 Conclusions

In this paper, we have provided a model for analyzing an access bandwidth market, where two providers offering different types of contracts compete in a price game with the goal of maximizing their profits. One of the providers offers static contracts under which customers buy the necessary bandwidth at the beginning of a long-term period. The other provider offers dynamic contracts where each customer can express his demand at the beginning of smaller-scale periods (slots). Both providers seek to maximize their profits by posting prices at the beginning of time and keeping them fixed through the long-term period.

We have provided a model to capture the customers' varying demand for bandwidth of the customers, allowing them to have different levels of demand in each slot. We have calculated the reaction curves for the case of constant provision costs and we have shown the existence of an equilibrium point. Then considering the implications of the market size in the provision cost, we have tried to capture the effects of multiplexing in the cost structure of the dynamic provider. With the use of the effective bandwidths theory, we have extended our initial problem and have provided the new maximization problems. Through numerical analysis, we have plotted the reaction curves and we have investigated how they are affected by the market size. Furthermore, we have studied the conditions under which a provider of dynamic contracts can participate in the market, the profits he makes and the market segment he obtains.

As future work, we would like to investigate how different models for demand affect our results. More precisely, in our initial model of constant costs, we would like to see how the reaction curves are affected under different distributions for p. In the model of multiplexing-dependent provision cost for the dynamic provider, we would like to examine if and how we can include a penalty for the provider of dynamic contracts when not fulfilling the customer's bandwidth requirements, i.e. the probability defined by the IOP for the inventory overflow. Finally, the properties of *mixed contracts* must be studied in the case of a competitive environment. Mixed contracts are a combination of static and dynamic contracts, under which the customer can buy at a price a a static amount of bandwidth for the whole longterm period (static part), but has also the ability to buy extra bandwidth at a price b, at the beginning of each slot (dynamic part). In our previous work we have examined some of their properties in a monopoly setup.

Acknowledgments

This work is part of the project "Economic and Architectural Aspects for Services of Differentiated Quality in high-speed Broadband and Wireless Access Networks", co-funded by the "European Social Fund & National Resources - EPEAEK II - IRAKLITOS Fellowships for research of Athens University of Economics and Business". This work has been supported by the EuroNGI Network of Excellence (EU-IST -507613) and has also a partial support of the IST project GridEcon (IST-2006-033634) funded by the EU.

References

[Binmore 1991] Binmore, K.: "Fun and Games - A Text on Game Theory," Lexington, MA: D. C. Heath and Co., Eds., 1991.

[Courcoubetis and Weber 2003] Courcoubetis, C., Weber, R.: "Pricing Communication Networks: Economics, Technology and Modeling," Chichester, Wiley, 2003.

[Courcoubetis et al. 2006] Courcoubetis, C., Soursos, S., Weber, R.R.: "Pricing Resources on Demand," *1st IEEE International Workshop on Bandwidth on Demand,* 2006, pp. 12-18.

[Davies et al. 2004] Davies, G., Hardt, M., Kelly, F.: "Come the revolution-network dimensioning, service costing and pricing in a packet switched environment," *Telecommunications Policy,* 2004, v.28, pp. 391-412.

[Foros and Hansen 2001] Foros, O., Hansen, B.: "Competition and compatibility among Internet Service Providers," Information Economics and Policy, 2001, vol. 13, issue 4, pp. 411-425

[Gibbens et al. 2000] Gibbens, R., Mason, R., Steinberg, R.:"Internet Service Classes under Competition," IEEE Journal on Selected Areas in Communications, 18(12), December 2000, 2490-2498

[Mason 2000] Mason, R.: "Simple Competitive Internet Pricing," *European Economic Review,* 44(4-6), 2000, pp. 1045-1056.

[Rabbat and Hamada 2006] Rabbat, R., Hamada, T.: "Revisiting Bandwidth-on-Demand Enablers and Challengers of a Bandwidth Market," *Network Operations and Management Symposium,* 2006, NOMS 2006, 10th IEEE/IFIP, pp. 1-12.

Cost Analysis of Current Grids and its Implications for Future Grid Markets

Marcel Risch, Jörn Altmann

International University in Germany, School of Information Technology,
Campus 3, 76646 Bruchsal, Germany
{Marcel.Risch, Jorn.Altmann}@i-u.de

Abstract. Commercial Grid markets have been a topic of research for many years. Many claims about the advantages of trading computing resources on markets have been made. However, due to a lack of Grid computing offerings, these claims could not be verified. This paper analyzes the question whether using the Grid is financially advantageous, using the Amazon.com EC2 service as a reference. To perform this analysis, the costs of computing resources in different usage scenarios are calculated, if Grid resources and in-house resources are used. The comparison of the costs reveals that while the Grid is cheaper in the short term, it is not a good investment in the long term and, thus, the existence of a Grid economy will not lead to an end of ownership but rather to a reduction in in-house resources and more efficient resource usage.

Keywords: Commercial Grids, Grid Computing, Business Models, Cost Modeling, Capacity Planning, Grid Economics, Utility Computing, Markets.

1 Introduction

Commercial Grids have been a focal point of research for many years. The idea of selling idle computing resources on a market for computational power has been advocated since the early 1960s. With the advent of the Internet and the Internet economy, this idea once again received attention during the last years. The advantages of Grid markets have been emphasized with various claims that could not be validated, as a Grid economy did not exist. However, such an economy has now started to develop with the introduction of a number of cluster or cloud computing providers, who sell compute resources on a pay-per-use basis. Using these offers as a basis, we are now able to validate some of the frequently made claims about commercial Grids.

In this paper, we will focus on four claims. They address the financial advantages that companies would gain from using a commercial Grid. Since companies are, in general, seeking ways to gain competitive advantage or to lower their operational costs, the financial advantages of Grids play a major role in promoting Grid usage. Therefore, we will analyze the following four claims:

– *Claim 1:* Companies can reduce the staff for maintaining resources. This idea has been propagated in research and commercial circles [1][2][3].

- *Claim 2:* Companies have large computational power available at their fingertips on a pay-per-use basis [4][5].
- *Claim 3:* Companies do not have to purchase the resources and, thus, have no cost of ownership [2][3][4], which is significant for high-performance computing resources.
- *Claim 4:* The advantages of commercial Grids are the reduction in cost [4][5].

Although Claim 1 is difficult to validate for all enterprises, we believe that, due to the difficulty of using Grid resources, it is highly unlikely that an increased Grid usage will result in major savings from personnel reductions in medium-sized companies. There are a number of issues that indicate that the headcount in the in-house IT support staff will remain unchanged. Firstly, the software running on a Grid resource must be maintained and monitored in the same way as the software running on in-house resources. Secondly, the in-house staff must be able to handle many different virtualization tools, such as Xen [6], VMWare [7]. For each of these tools used in a Grid market, the in-house staff must maintain and create the correct images. Since there are as of now no support tools available, the in-house staff must be knowledgeable in may different virtualization tools. Thirdly, any company using the Grid has to perform a detailed cost-benefit analysis to determine whether using the Grid is more cost effective than purchasing in-house resources. Since this analysis requires intimate knowledge of all applications, hardware, and the skill to predict the load levels, experienced staff is needed.

Due to these reasons, we do not believe that the size of in-house IT staff for medium-sized companies can be reduced due to increased Grid usage. Small enterprises, on the other hand, will not only require the computational power, they will also need some software to run on these computers. For these companies, the Grid is more interesting if it offers Software-as-a-Service and not just pure computing power. For large companies, the cost savings through Grid usage are very little. Large companies already benefit from the economies of scale in the operation of their IT resources. IT resources of large enterprises are organized in a few data centers, supported by a sufficiently large number of in-house IT staff. Therefore, any outsourcing of the data center service (i.e. using the Grid) could not result in significant cost savings. Since no type of enterprise is expected to reduce its IT support staff headcount through the use of Grid computing, we will not include the personnel costs in the following parts of this analysis.

Continuing with Claim 2, we can state that this claim is obviously true: All currently existing Grid resource providers have a pricing structure, in which the customer only pays for the computational power used. Furthermore, while there are some limits imposed on the number of resources available, in general, these limits are fairly broad and should not pose any difficulties for users.

Claim 3 is also trivial to verify. Since Grid resources are not purchased but are rather rented to the buyer, the buyer has no costs of ownership. Since Grid resources are not owned by the purchasing company, the purchasing company does not have any costs of ownership.

To perform the analysis of Claim 4, we need to consider the market structure, the type of resources sold and the size of the enterprise using the Grid. Based on this information and evidence gathered in the existing commercial Grid environment, we

will determine during the remainder of this paper whether this claim can be supported.
This paper is structured as follows: In section 2, we analyze the potential Grid users, the types of resources available, and the current market structure. In the third section, we present some basic cost information for Grid and in-house resources. In addition, we characterize three companies that are used for comparing Grid and in-house resource costs. These companies need to acquire additional resources, which have the characteristics of one basic instance of an Amazon EC2 resource [8]. In section 4, we present a case-by-case cost analysis for different Grid usage scenarios. In the fifth section, we will analyze the results of the case study and draw some conclusions about the structure of the future Grid market. Finally, we conclude by presenting some open items, which can be explored in future research.

2 Analysis Framework

2.1 Potential Grid Users

For our analysis, we assume that commercial enterprises mainly use the Grid. These enterprises can be categorized as follows: i) home offices; ii) small enterprises; iii) medium-sized enterprises; and iv) large enterprises. The definitions are standardized in the European Union [9]. In addition to this, we also define the companies in terms of their IT expertise. In general, the smaller the company, the less IT expertise it has. In other words, home offices and small enterprises have less IT expertise than medium-sized or large enterprises. Therefore, their needs for IT solutions differ. Thus, home offices and small enterprises need more complete solutions (e.g. Software-as-a-Service) for their IT needs than medium-sized or large enterprises. Large enterprises, which can perform any kind of IT investment and already benefit from the economies of scale (which Grid computing promises), would not get any additional benefit from participating in a commercial Grid.

Therefore, for our analysis, we only consider medium-sized enterprises. Those companies are characterized by restricted budget for IT investments, and the existence of an IT department.

2.2 Resource Types Available on a Grid

In general, any type of compute resource can be sold on a Grid market. However, for the purpose of the analysis, a classification of those resources helps highlighting the characteristics of those resources. Our classification of computing resources resulted in the following four groups:

- High Performance Computers (HPC): Supercomputers for specialized tasks and compute-intensive applications.
- Server clusters: A number of servers, which are located in the same facility and interconnected to ensure high communication speeds between the individual

servers. They can be used for high-performance computing as well as for monolithic applications as the computing resources of the next group.
- Servers: Individual servers for running monolithic applications.
- Desktops: Individual workstations for employees.

In this paper, we will focus solely on individual servers, since the existing computing cloud offerings (e.g. Amazon EC2 service) is aimed at companies requiring additional servers. This also makes the comparison between in-house resources and Grid resources easier, since prices for in-house resources can be easily obtained from various hardware manufacturers.

2.3 The Market Structure

The structure of the current Grid market is an oligopoly. We have only a few large providers in the market, such as Amazon.com EC2 [8], Sun Grid [10], and Tsunamic Technologies [11]. Because of this market structure and slight differentiation of their services, they can set their prices such that it maximizes profits.

An alternative market structure would be characterized by complete competition between resource providers, who sell their excess resources on a cost basis. In such a market, prices would be generally lower due to competition and the only price fluctuations would be caused by high demand. The demand for resources is higher than the available resources on the Grid.

In this paper, we will focus on the current market and more specifically, on resources obtained from Amazon.com's EC2 service. This provider was chosen for a number of reasons. Firstly, it started its resource sales shortly after the advent of the GridEcon Project [12]. Secondly, the pricing structure is very well described, making it easy to calculate the prices for different usage scenarios. Thirdly, the provider was chosen for its clear specification of the virtual machines, thus ensuring that equivalent servers for in-house installation can be found easily. Lastly, the Amazon EC2 service was chosen due to its popularity: According to [13], this service is now used by about 60,000 customers and generates a revenue of about $131 million.

3 Methodology and Data Collection

To determine in which cases the Grid is cheaper than in-house resources, we will use three companies, called C_1, C_2 and C_3, which require additional resources in the form of a single server. Furthermore, we will assume a linear growth of costs for all companies, i.e. if the price for a single server is P, then the cost for n servers will be n*P. Economies of scale are neglected, since it is difficult to estimate both the point at which they set in and the magnitude of the discount.

The three companies will obtain their resources as follows: Company C_1 will purchase its server for in-house installation expensively. Company C_1 is assumed to be a small company with little purchasing power. Company C_1 has higher costs than company C_2 which also obtains its resources for in-house installation. Company C_2 is

assumed to be bigger and, therefore, has higher purchasing power. Company C_3 will purchase resources on the Amazon EC2 service.

Since company C_3 uses the Amazon EC2 resources, the resources used by the other companies should be comparable. In particular, we will assume that all servers have at least a 2GHz, single-core CPU, at least 2GB of main memory and a hard disk with at least 200GB storage. To match the requirements, company C3 will purchase another 40GB of storage from Amazon's S3 service [14].

The prices for the resources used by companies C_1 and C_2 were obtained using the online tools of Dell [15], Gateway [16], and HP [17]. Based on the prices found, company C_1 has been assigned a server, which is 25% more expensive than the most expensive model. This price was chosen to ensure that C_1 has the highest costs and thus, has the largest incentive for using the Grid. Company C_2, on the other hand, pays the average price for all resources, thus ensuring that the price paid by C_2 is realistically chosen.

The prices that company C3 faces have been obtained from the Amazon EC2 Web site. The actual prices for all resources are described in chapter 4. Since the Amazon.com cost structure emphasizes usage times, we will assume that a server is used continuously for 30 days. This will be the basis for the comparison in chapter 5.

Since the costs of Amazon.com's EC2 depend on the actual usage of the resources, we have to introduce usage scenarios which take the actual usage into account. We have decided on the four scenarios listed below. These were chosen because they illustrate different generic usage patterns that may be encountered by SMEs using the Grid. These scenarios also illustrate the effect the pricing structure has on the overall Grid cost. The term "upload bandwidth" refers to the data transferred out of the Amazon.com EC2 service and the term "download bandwidth" refers to the data transferred into the service.

- *Scenario 1*: Update Server: The server uses a lot of upload bandwidth and little download bandwidth. Such a server would be used for companies with many customers
- *Scenario 2*: Backup Server: The resource uses a lot of download bandwidth and less upload bandwidth. This type of server would be used for off-site backups for important data.
- *Scenario 3*: Computational Server: The resource uses little bandwidth as it is mainly used for computations.
- *Scenario 4*: Medium-Sized Enterprise Web Server: A server that is barely used but hosts a vital program for the company, such as a Web server.

There is one additional alternative to using commercial Grids: Virtual Private Server (VPS) hosting. There are a number of providers of this type of service; however, the resources offered are geared more towards web hosting rather than computation. This is made obvious by the lack of resource specification when it comes to processor speeds. Instead, customers are attracted by the amount of storage offered and the main memory size.

We have compared a number of VPS providers, such as EMC [18], InMotion [19] and Yourserving.com [20]. We have found that the resources most comparable to the ones offered by the Amazon.com EC2 service cost between $90 and $170 per month, depending on the subscription length and the provider. Since these costs are significantly above the costs for in-house resources and since the bandwidth

allowances are sometimes severely restricted, we have decided that this type of service is not an adequate replacement for in-house resources or for Amazon.com EC2 resources. Therefore, we have ignored this service type in our analysis.

4 Cost Calculation

Under normal circumstances, resources will be written off after three years using normal depreciation rules. This means that every month, a depreciation cost is incurred which is added to the other monthly costs. In our cost calculation, we will simplify matters by assuming that depreciation is not used, but rather that the entire cost of an in-house resource has to be paid upfront. This allows the reader to see when the Grid usage costs reach or exceed the costs for in-house resources.

In this section, we introduce the costs that the three companies face. In addition, we will calculate the monthly costs for each of the two companies that use in-house resources.

4.1 Company C_1

As we have stated earlier, company C_1 obtains its resources expensively. The following is a list of costs that C_1 will have to pay for its resources.

- New server: From our research, we have found that an expensive new server costs no more than $650. We will assume that C_1 will have to pay $800, which is more than 25% more than the highest price we found.
- Electricity: According to the Energy Information Administration (EIA) [21], the electricity cost for commercial enterprises is at most 14.65 ct/KWh in the contiguous US. For C_1, we will assume an electricity cost of 20.00 ct/KWh. To calculate the monthly electricity usage costs, we need to determine the power consumption for the server. Power supplies usually range between 200W and 500W; since the resource used in our calculation is a server which does not require power-hungry components, we will assume that the power supply is in the middle of this range. Therefore, we chose a 350W power supply which means that the server uses 350W/h. This means that one hour of operating the server costs $0.350 KWH * 0.20 \frac{\$}{KWH} = 0.07\$$.

In the first month, the company will have to pay both the server and the electricity. The costs are shown in Table 1. For the following months, only the electricity costs must be paid which means that the monthly costs are at $50.40.

Table 1. First month costs for C_1

	Quantity	Price	Total ($)
Hardware Purchase	1	800 $	800.00
Electricity	720 h	0.07 $/h	50.40
Total (first month)			850.40

4.2 Company C_2

A similar calculation has to be performed for company C_2. This company is able to obtain its resources and electricity cheaper than company C_1, thus having a competitive advantage without the Grid. The costs of C_2 are divided as follows:

- New server: The average server price for the Amazon EC2-type server was about $500. We will assume that company C_2 paid this price for its resources.
- Electricity: Using the EIA table again, we decided to use a more realistic electricity price. Since a large number of IT companies is located in the California, we decided to the use the average commercial electricity price for 2007 as a reference. At time of writing of this paper, the price was 12.76 ct/KWh, which was rounded to 13 ct/KWh for easier computation. Using an online power calculator [22], we determined that a server would use about 200W. This means that one hour of operating the server costs $0.200KWH * 0.13\frac{\$}{KWH} = 0.026\$$.

Using this information, we can now calculate the usage costs for the first month for company C_2. This information is given in Table 2.

Table 2. First month costs for C_2

	Quantity	Price	Total ($)
Hardware Purchase	1	500 $	500.00
Electricity	720 h	0.026 $/h	18.72
Total (first month)			518.72

We can see that company C_2 has much lower costs than company C_1. In addition, it should be noted that the monthly costs are less than half of the costs incurred by C_1, namely only $18.72.

4.3 Company C_3

Finally, the costs for company C_3 need to be introduced. Since C_3 uses the Amazon services EC2 [8] and S3 [14], the total cost incurred for each month depends on the actual usage. In Table 3, the costs for the various items are shown.

Table 3. Costs for using Amazon.com EC2 and S3

Item	Cost	Restrictions
Hourly cost	0.10 $/CPU-hr	None
Data Transfer In	0.10 $/GB	None
Data Transfer Out	0.18 $/GB	First 10 TB/month
Data Transfer Out	0.16 $/GB	Next 40 TB/month
Data Transfer Out	0.13 $/GB	Over 50TB/month
Hard Disk Space	0.15 $/GB-month	Each GB over 160 GB

5 Cost Comparison of Each Scenario

In this section, we determine the costs incurred by using the Grid in each of the four scenarios and compare it with the cost of in-house purchases. These scenarios are defined in the form of usage characteristics.

5.1 Scenario 1: Download Server

The download server uses a large amount of upload bandwidth and very little download bandwidth. As a basis for our calculation, we used some data from the SecondLife Blog [23]. We assumed that the download server would be used heavily for four days and then be used less for the remainder of the month. Since the blog referred to 70GB of downloads per hour for almost one day which was then followed by several days of 30GB per day, we decided on the following upload quantities: 70GB for the first day, 30GB for the following 3 days and 3.5GB for the remaining 26 days:

$$70\frac{GB}{hr} * 24hr + 30\frac{GB}{hr} * 24hrs * 3 + 3.5\frac{GB}{hr} * 624hr = 6024GB$$

The result was rounded to 6000GB for the 30 day period to simplify the calculation. The entire cost for the first 30 days of operating a Grid resource is calculated in Table 4.

Table 4. Download server costs

	Quantity	Price	Total ($)
CPU-hrs	720	0.10 $/h	72
Hard Disk Space	40	0.15 $/GB	6
Upload Data	6000	0.18 $/GB	1080
Download Data	100	0.10 $/GB	10
Total			1168

It can be easily seen that the Grid in this case is extremely expensive, mainly due to the high upload costs. Comparing this value with the prices obtained by companies C_1 and C_2, we can see that C_3 pays 125% more than the amount paid by C_2 and 37% more than C_2.

However, in this scenario, we assume that C_1 and C_2 have sufficient bandwidth to satisfy the download requirements. Since this amount of bandwidth is usually not available for medium-sized companies, both C_1 and C_2 would have to purchase additional bandwidth. In order to support 70GB/day in uploads; they would need about 14 lines of an AT&T 6Mbit download line service [24]. This costs $60 per line and therefore, the total cost for 14 lines would be $840 per month. Alternatively, a Verizon 15Mbit upload line could be purchased for about $240 per month [25].

In both cases, the monthly cost for companies C_1 and C_2 would increase. However, in the long-term the Grid would still be more expensive. This can be demonstrated by showing the cost graphs of all companies. In Figure 1, we show the cost graphs over time if both C_1 and C_2 use the more expensive Internet access service. If the

companies would use the less expensive option the slope of the curve would be even lower.

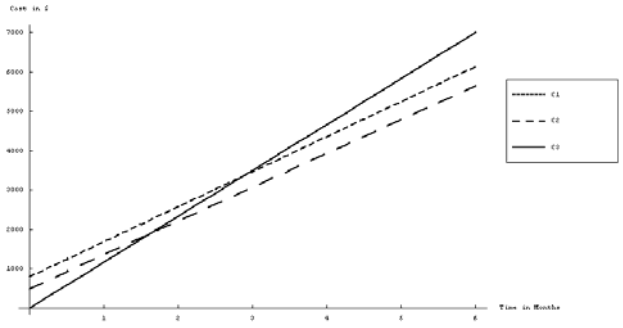


Figure 1. Price comparison with expensive internet

As we can see, even with the expensive Internet access, the costs of the Grid are higher than for in-house resources after three months. It should be noted however, that the Internet prices require a one-year subscription. If the company requires the high download bandwidth for one month only, then the Grid would be much cheaper, since the Internet connection would cost at least $2900 for a one-year subscription.

5.2 Scenario 2: Backup Server

A backup server has a high number of downloads and a low number of uploads, assuming that the data stored in the backup server is rarely needed. Since the upload bandwidth is not used as much as in the first case, the companies using in-house resources would not have to resort to purchasing additional Internet connectivity. For company C_3, we will assume that it performs uploads of 500GB every month. This corresponds to loosing two complete sets of data and making some minor corrections. Furthermore, we assume that the company downloads about 3000GB. This corresponds to backing up 100GB every day and replacing copies after two days. The monthly cost calculation can be seen in Table 5.

Table 5. Monthly costs of a backup server

	Quantity	Price	Total ($)
CPU-hrs	720	0.10 $/h	72
Hard Disk Space	40	0.15 $/GB	6
Upload Data	500	0.18 $/GB	90
Download Data	3000	0.10 $/G	300
Total			468

In this case, the Grid is cheaper than in-house resources in the beginning as well. However, as the monthly costs are much higher when using the Grid, company C_3 soon pays more than the companies purchasing in-house resources. After about 1.5

months, company C_2 will pay less than company C_3; after 2.5 months company C_1 will pay less than company C_3. This development is shown in Figure 2.

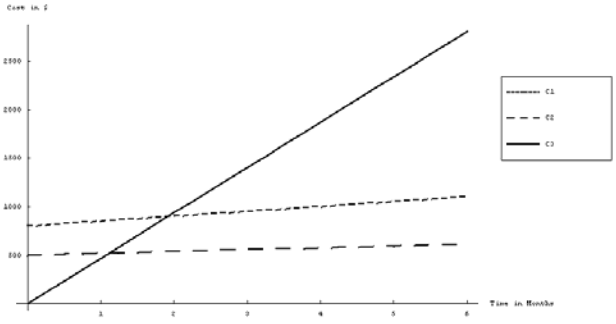


Figure 2. Price comparison for backup servers

The figure illustrates how much more expensive the Grid is in the long run. However, it should be noted that this calculation is only valid if the company who uses in-house resources can host the new server. If the new server has to be hosted at a different location, which is not owned by the company, or if such a location has to be built or bought, then the Grid will be cheaper, since the costs for the new location will be much higher than the monthly Grid costs.

5.3 Scenario 3: Computational Server

So far, we have only examined resources that require large amounts of bandwidth. However, bandwidth is one of the main cost drivers of the Amazon EC2 service. Therefore, we will now focus on a server which requires less bandwidth and is largely used for compute-intensive tasks. We assume that the server requires 100GB upload and 100GB download, since this server may be part of a computationally large workflow where the individual subjobs transfer data between each other. The monthly Grid costs are detailed in Table 6.

Table 6. Computational Server monthly costs

	Quantity	Price ($)	Total ($)
CPU-hrs	720	0.10 $/h	72
Hard Disk Space	40	0.15 $/GB	6
Upload Data	100 GB	0.18 $/GB	18
Download Data	100 GB	0.10 $/GB	10
Total			106

Although the expenses for Internet access are still significant, the monthly cost is much lower than that of the previous scenario. Even when using less bandwidth, the Grid is still more expensive than in-house resources in the medium-term. The cost difference between the bandwidth-intensive servers and this server is reflected by the

fact that the breakeven point between Grid resources and the in-house resources has been moved to a later date: for company C_2, the breakeven is reached after slightly less than six months, for company C_1 the breakeven is reached after a little more than 14 months. This is illustrated in Figure 3 below.

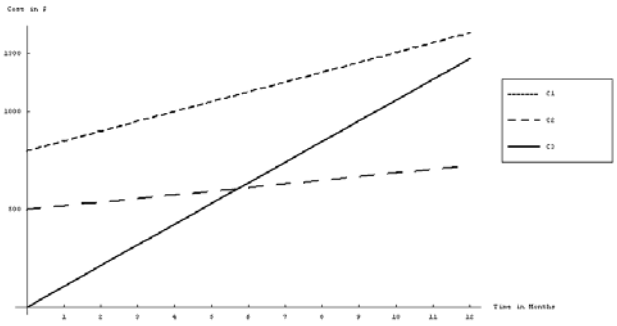


Figure 3. Price comparison for computational servers

5.4 Scenario 4: SME Web Server

In this case, we assume that a small, little-known company uses the Grid to set up a Web server. Since the company is not known, there will be very little traffic on the server, and therefore, these will be almost no bandwidth used. However, since Amazon.com charges the user for each started GB of bandwidth used, we will take some minimal traffic into account. During the first month, we will assume some higher download usage, since the machine image will have to be transferred. For the subsequent months, we will assume that only web traffic will be incurred. For this traffic, we will assume that each web page has a size of about 100 KB and that the ten pages are request per day. This means that about 30MB of data transferred out of Amazon.com.

Furthermore, we will assume that the company will not purchase additional hard disk space on Amazon's S3 service. Only the costs for the subsequent months are shown in Table 7, the costs for the first month are only slightly and can therefore be neglected.

Table 7. SME web server first month costs

	Quantity	Price	Total ($)
CPU-hrs	720	0.10 $/h	72.00
Upload Data	1	0.18 $/GB	0.18
Download Data	1	0.10 $/GB	0.10
Total			72.28

These costs are significantly lower than those for of the previous scenarios. Thus, the Grid price in this case is much more competitive than in the previous cases. This fact is illustrated in Figure 4 below.

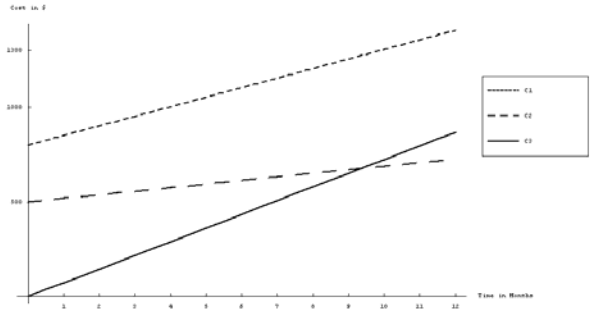


Figure 4. Cost comparison for a SME web server

Compared to the resources bought by company C_2, the Grid becomes more expensive after about 9 months. Since company C_1 has higher costs, the breakeven will not be reached until about month 35. Therefore, we can conclude that the Grid is cheaper in the medium- to long-term if the in-house resources are very expensive. However, if the in-house resources are cheap, then the Grid is only cheaper in the medium-term.

6 Analysis

This section will consist of two parts: in the first part, we will discuss the remaining claim using the information provided in section five. In the second part, we will determine the implications that the analysis results have on the future Grid market.

6.1 Claim Analysis

In the introduction to this paper, we have given four claims about Grid economies. Of these we have already addressed three claims, namely the claim that companies have large computational power available at a pay-per-use model, the claim that companies can reduce their in-house staff, and the claim of no cost of ownership.

In the last claim, it was asserted that the Grid reduces the cost for hardware resources. This claim has been the starting point for the detailed cost analysis in the previous section. From our calculations, we can state that using the Grid is not always cheaper than using in-house resources. In fact, every company must determine for itself at which point the Grid becomes too expensive. In general, the cost-effectiveness of the Grid depends on two parameters: the usage duration and the usage intensity. Using these parameters, we can enumerate some cases, in which the Grid usage would be advantageous:

– To cover short, infrequent demand peaks. These peaks should not occur more often than once every several months, or once every year. The peaks last for a few weeks at most.
– If the data backup should be made in a physically different location, which cannot be afforded by a company otherwise.
– Lightly used resources over a short to medium-term period.

Since there are cases, in which the currently existing Grid market is not cheaper than in-house resources, we can conclude that the existence of a Grid economy will not lead to an end of ownership. But, companies will be able to reduce their resource infrastructure by covering infrequent usage peaks with Grid resources. However, regularly occurring peaks must still be provided for using in-house resources. Consequently, there will still be in-house resources that remain idle for some periods of time. Two general statements can be made, based on our calculations:

– *For heavily used resources*: If the resource requirements exceed the in-house capacity for less than two months during a depreciation period of three years, then the current Grid market is cheaper. Therefore, given the current resource prices, the usage duration is the main decision factor for which resources should be bought. If the resource is used for less than 6% of the three years (depreciation period), it should be bought on the Grid.
– *For less heavily used resources*: If the resource requirements exceed the in-house capacity for less than six months during the three year depreciation period, then the current Grid market is cheaper. Therefore, if the resource is used for less than 17% within a period of three years, it should be bought on the Grid.

6.2 Implications for the Future Grid

We have determined that companies will still have excess of in-house resources. Therefore, a solution needs to be found as to what can be done with these resources when they are idle. There are two courses of action open to companies: They either turn off the resources to conserve electricity and thereby reduce expenditures, or, they sell the excess resources on a Grid market for commodity goods. The first option is sub-optimal, since only the electricity costs are reduced. Since electricity at this point remains cheap, the savings will be fairly low. The second course of action, on the other hand, will allow companies to recoup most of their costs, including maintenance and depreciation costs. This added income would allow the company to leave its resources switched on while at the same time ensuring that no money is lost due to idle resources.

A Grid market for commodity goods, in which companies can sell their idle resources, would be characterized by intense competition between resource sellers. The advantages of such a market are numerous: Due to the intense competition, the prices would be lower than in the current Grid market, which is a seller's market. This lower price would, in turn, encourage buyers to purchase more Grid resources, since the difference in cost between the Grid resources and their in-house resources is relatively small. It would also lower the barrier of entry to the Grid for new users. In addition, Green IT objectives are met, since resources are used to their full capacity and, therefore, resources rarely sit idle any longer.

A Grid market as described above has to fulfill some requirements: Firstly, it has to sell commodity goods, which are comparable and substitutable. Therefore, the market allows for a competitive market environment. Secondly, the Grid market has to be able to manage many providers and buyers in a single platform. It must be able to handle a large volume of trades and store large amounts of data about these trades. Due to the competition, resource providers will use marginal pricing for their resources to remain competitive. Only congestion which is caused by short term high demand peaks will cause high prices.

For such a market to operate smoothly, some support services need to be developed. These services are especially important for companies that have little or no Grid expertise. This idea has been at the heart of the GridEcon project which has developed a framework to support services for SMEs with little Grid expertise. These services include various brokers (e.g. Risk Broker, Workflow Broker, and Insurance Broker) as well as services such as a Capacity Planning Service. The goal of these services is to simplify the transition to the Grid and its usage as much as possible.

7 Conclusion

In this paper, we have discussed four claims about Grid computing, and analyzed one in detail. We have found that the Grid is not always cheaper than in-house resources. Since, at present, only few Grid resource providers exist in the market, they can easily generate profits. Therefore, the effects of the economies of scale are negated. Therefore, any company considering the use of Grid resources should carefully calculate whether the Grid is actually cheaper than in-house resources.

From the analysis of the costs of the current Grid (which is a set of data centers of servers), we have determined that the existence of a Grid will not lead to an end of ownership but will lead to a decrease in over-provisioning of computing resources. We expect that rare demand peaks will be covered using Grid resources.

Since, under the current market structure, companies still have to over-provision, they will have to face the question of what to do with idle resources. Selling these resources on a Grid market is the best option, since all incurred costs can be recouped. If many companies sell their idle resources on a market, this will lead to strong competition, which will force prices to remain low unless there is a severe resource shortage. The low prices will attract more buyers, thereby increasing supply and demand. During times of high demand, companies may even be able to make small profits due to the increased prices they can charge.

The workings of this competitive market need to be studied further, with special attention paid to price setting, the price development over time, the actions taken by resource sellers and buyers, and the effects these actions have on the market. This also leads to the question of how companies will act and react to price fluctuations.

In addition, the analysis performed in this paper can also be repeated for other resource types, such as differentiated goods. The results could form the basis for a Grid markets for differentiated goods.

8 References

[1] Southern Partnership for Advanced Computational Infrastructures (SPACI), http://www.spaci.it/content.php?loc=projects&pg=prj.php&cat=gm&id=10

[2] Yeo, C.S., Buyya, R., Assunção, M.D., Yu, J., Sulistio, A., Venugopal, S., Placek, M.: Utility Computing on Global Grids. In: Bidgoli, H. (eds.) The Handbook of Computer Networks, chapter 143. John Wiley & Sons, New York, USA (2007).

[3] Fujitsu Siemens Computers, http://www.fujitsu-siemens.it/it_trends/grid_computing.html, 2008.

[4] Altmann, J., Courboubetis, C., Darlington, J., Cohen, J.: GridEcon – The Economic-Enhanced Next-Generation Internet. In: GECON 2007, LNCS vol. 4685. Springer, Heidelberg (2007).

[5] Altmann, J., Ion, M., Mohammed, A. A. B.: Taxonomy of Grid business models. GECON 2007, LNCS 4685, Springer, Heidelberg, 2007.

[6] XenSource, Inc., http://xen.org/, 2008.

[7] VMware, Inc, http://www.vmware.com/, 2008.

[8] Amazon Elastic Compute Cloud (Amazon EC2), http://www.amazon.com/gp/browse.html?node=201590011, 2008.

[9] Enterprise – SME Definition, http://ec.europa.eu/enterprise/enterprise_policy/sme_definition/index_en.htm, 2008.

[10] Sun Grid, http://www.sun.com/service/sungrid/index.jsp, 2008.

[11] Tsunamic Technologies Inc., http://www.clusterondemand.com/, 2008.

[12] GridEcon, http://www.gridecon.eu, 2008.

[13] Schonfeld, E.: Who Are The Biggest Users of Amazon Web Services? It's Not Startups. http://www.techcrunch.com/2008/04/21/who-are-the-biggest-users-of-amazon-web-services-its-not-startups/, 2008.

[14] Amazon Simple Storage Service (Amazon S3), http://www.amazon.com/gp/browse.html?node=16427261, 2008.

[15] Dell, http://www.dell.com/, 2008.

[16] Gateway, Inc., http://www.gateway.com/, 2008.

[17] Hewlett-Packard Development Company, L.P., http://www.hp.com/, 2008.

[18] EMC Telecom Corporation, http://www.emcwebhosting.com/vps_hosting_extreme_linux.php, 2008.

[19] InMotion Hosting, Inc, http://www.inmotionhosting.com/vps_hosting.html, 2008.

[20] yourserving.com, http://yourserving.com/vps_server, 2008.

[21] Energy Information Administration, Average Retail Price of Electricity to Ultimate Customers by End-Use Sector, by State, http://www.eia.doe.gov/cneaf/electricity/epm/table5_6_b.html, 2008.

[22] Journey Systems, LLC., Power Calculator, http://www.journeysystems.com/?powercalc, 2008.

[23] Linden Research, Inc., SecondLife Blog, http://blog.secondlife.com/2006/10/26/amazon-s3-for-the-win/, 2008.

[24] AT&T, https://swot.sbc.com/swot/dslMassMarketCatalog.do?do=dslProductPage&offerId=90027&serviceType=DYNAMICIP, 2008.
[25] Verizon, http://www22.verizon.com/content/businessfios/packagesandprices/packagesandprices.htm, 2008.

Market Mechanisms for Trading Grid Resources

Costas Courcoubetis[1], Manos Dramitinos[1], Thierry Rayna[2],
Sergios Soursos[1], and George D. Stamoulis[1]

[1] Network Economics and Services Group (N.E.S.), Department of Informatics,
Athens University of Economics and Business (AUEB),
76 Patision Street, Athens, GR 10434, Greece
{courcou,mdramit,sns,gstamoul}@aueb.gr
[2] Internet Centre, Imperial College London, South Kensington Campus,
180 Queen's Gate, London SW7 2AZ, United Kingdom
t.rayna@imperial.ac.uk

Abstract. There has been recently an increasing interest in Grid services and economic-aware Grid systems both in the industry and the academia. In this paper we specify a market for hardware providers and consumers interested in leasing Grid resources for a time period. Our approach comprises a stock-market like mechanism that enables the trading of computational power on the basis of a spot and a futures market. The spot market comprises a pair of bid and ask queues. This grid market is more complicated than the standard spot/futures markets of storable commodities, because the computational service traded in our case comprises of resources that are perishable, and has both quantity and duration specified in terms of a time interval. This is an important feature of our market mechanism, complicating considerably the trading algorithms that we develop and assess in this paper.

Keywords: Market mechanism, grid market, bid and ask, spot, futures.

1 Introduction

Motivated by the electrical power grids and the time-sharing computational systems of the past, there has been an increasing interest in Grid services over the past years. In order to materialize the virtualization and wide-scale sharing of computational resources, a variety of related business models regarding utility computing and software on demand have been developed, while economic-aware Grid systems have become increasingly popular both in the industry and the academia [1]. A wide variety of related market systems have been proposed; these are based on fixed prices, bartering, negotiations or auction models for leasing "Grid contracts". A detailed overview and presentation of these economic-aware grid systems and architectures can be found in [2] and [3]. However, despite the various economic mechanisms that have been proposed as candidates to be adopted in a Grid market, very few efforts have been made to fully specify the design of a market that is tailored to the Grid products and services. Indeed most proposals neglect taking into account the fact that both the resource type and the time dimension are of significant importance for the perishable Grid resources.

J. Altmann, D. Neumann, and T. Fahringer (Eds.): GECON 2008, LNCS 5206, pp. 58–72, 2008.
© Springer-Verlag Berlin Heidelberg 2008

In this paper, we specify a marketplace comprising all functionality for the leasing of computational service for a time period. It can serve as the core of the Grid economy. Customers (and hardware providers) interact through the marketplace, possibly by means of brokers, in order to lease (respectively offer) Grid resources. In Sect. 2 we outline a stock-market like mechanism and a corresponding system architecture that enables the trading of computational power on a spot market basis [4], as well as for selling futures contracts [5]. The underlying principle for this mechanism is that of a standard spot and futures market: All parties announce the maximum price they are willing to buy for and the minimum price they are willing to sell for. The spot bids (resp. asks) are put in the spot queue for bids (resp. asks). The futures are listed in the directory service of the futures market. All the compatible trades, i.e. when a bid is matched with a set of asks, are immediately executed. Note that matters are more complicated for our system's spot market than in standard spot markets of storable commodities, because in our case a the computational service is traded. This service is non-storable, because its specification includes both the quantity of resources and the relevant time interval. These matters are clarified in Sect. 3, 4 and 5, where we also introduce and assess an economically sound algorithm for matching bids and asks. Some additional important issues regarding matching that are left for future work, including the outline of a more sophisticated matching algorithm, are presented in Sect. 6.

2 The Marketplace and Its Architecture

So far, we have outlined the core functionality of the marketplace, i.e. the Grid Market, which is the main focus of this paper. It is worth noting that in order to be feasible for a *realistic* Grid marketplace to fully support the market mechanisms, a set of additional subsystems must be implemented as well. These subsystems are common among all existing (e.g. e-commerce) marketplaces, are not Grid-specific and both complement and support the core functionality of the Grid marketplace. Their detailed description is beyond the scope of this paper, which focuses on the presentation of a simple, fast and applicable, yet economically sound, Grid marketplace mechanism and the set of algorithms it comprises. However, we outline the marketplace system architecture (also depicted as Fig. 1) and highlight the subsystems functionality below for completeness reasons.

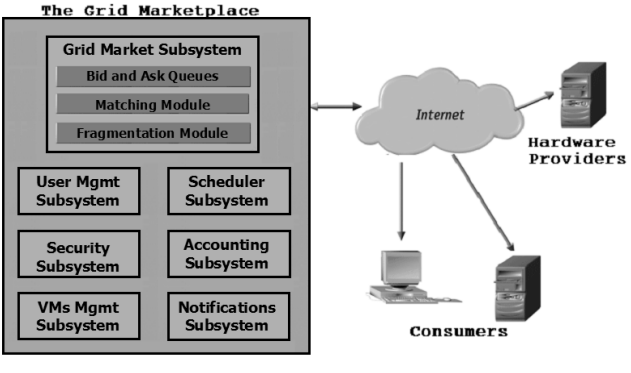


Fig. 1. The marketplace system architecture

User Management Subsystem: This subsystem is responsible for admitting users and providers into the marketplace. It uniquely authenticates and admits users/providers into the system, stores/checks their respective credentials and also interacts with the accounting/logistics and notifications subsystem.

Resource Management Subsystem: This subsystem is responsible for the management of the computational elements of the Grid marketplace and for the binding of the resources offered with a certain economic mechanism.

Security Subsystem: This subsystem enforces the marketplace's rules in the market transactions. It performs a wide variety of checks. For instance, it checks that the resources offered by the providers are indeed idle. It also interfaces with the Accounting and Logistics subsystems described below.

Accounting/Logistics Subsystems: Perform accounting and logistics management.

Directory Services Subsystem: This subsystem allows the organization and the advertisement of the leasing of resources. It is complemented with a search.

Notifications Subsystem: This subsystem is responsible for sending notifications to users. There are plenty of cases where this is desirable, such as: inform a bidder whether his bids are winning or not, or send reports to the virtual machines providers about their resource usage status and the respective revenue attained.

Scheduler Subsystem: This subsystem allows the execution of tasks at certain epochs, possibly periodically.

3 The Grid Market

First, we need to define the service that is to be traded in the grid market. Obviously, this must be suitable for the types of Grid applications currently existing or emerging. Hardware providers offer for leasing virtual machines (VMs) of different types that can be traded by means of different mechanisms [6], [7], [8]. It is expected that these resources be offered for a minimum desirable price and for certain time duration within a specific time interval, depending on the providers' supply constraints. Note that a virtual machine does not just correspond to a certain computational speed, but rather to an entire configuration of the hardware. This configuration is henceforth and throughout the remainder of this paper referred to as VM or unit of computation; these terms are used interchangeably.

An additional assumption of our model throughout the paper is that time is discretized in time slots. For simplicity of presentation reasons, the duration of this time slot in all the examples presented in the remainder of the paper is taken to be 1 hour, though in practice it would be set to a different value.

Customers are interested in accommodating their needs for computational power from the Grid market. They can achieve this by leasing some of the virtual machines that the providers make available in the Grid market. Depending on the nature of the tasks consumers may wish to execute, their demand can be expressed in a multitude of ways. A general type of "contract" is specified by means of a certain rate of computation for a specific time interval. For instance, this could be the case for a

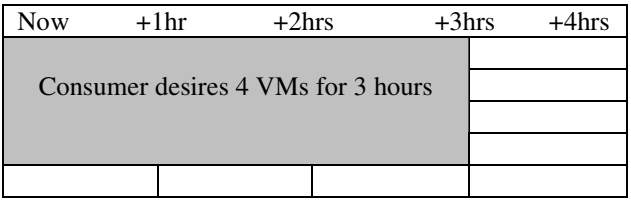


Fig. 2. A consumer's demand for 4 VMs for 3 hours is depicted by means of a rectangle

company's web server that leases Grid resources when it is critically loaded. This type of consumer need can be also graphically depicted by means of a rectangle (see Fig. 2): The height of the rectangle denotes the number of virtual machines required at any time of the interval, while width of rectangle denotes the amount of time for which these machines are needed.

Another type of contract could be specified by means of computational volume, i.e. a total number of VMs must be made available up to a maximum deadline constraint, so that a certain computationally intensive task is executed in time. As opposed to the previous case, only the total quantity of computational power is of interest, while the rate of computation provided at the various time epochs is not. This could be the case for a weather prediction program or a stock market data mining application that must be executed up to a deadline, i.e. the announcement of the weather forecast and the prediction of stock market prices before the markets open respectively. Note that the consumer needs in this case no longer correspond to rectangles, but rather to areas of rectangles, possibly with a maximum width (i.e. deadline) constraint. An extension of the Grid market mechanisms for this type of contracts is provided later in the penultimate section of this paper.

3.1 Bids

A bid in our system prescribes the resources required, which are specified by means of: a) the type and quantity of resources required, b) the starting time of the interval for using the resources, and c) the time duration of using the resources. It also specifies d) the price, expressed in €/min/unit and e) the time limit for which the bid applies. The latter is the maximum time at which the bid is considered to be valid. If this time is reached without the bid being matched, the bid must be removed.

The bid definition could also be complemented by a definition of whether or not the bid is atomic, i.e., it should be fully served by resources of a single provider. Atomicity may be the result of technological constraints on the possibility to switch execution environments. If such constraints are absent, economic theory suggests that the market should refrain from supporting atomic bids, due to the market power that large providers would obtain, which is not compatible with a bid and ask mechanism. In fact, even if the market would only support atomic bids, then it is likely that consumers would post-sale combine the resources of multiple such bids to obtain a more extended service Therefore, it is henceforth assumed that bids are non-atomic; more on this issue will be discussed later.

There are two types of bids in our system, namely future and spot bids. Future bids are the bids for which the starting and ending times are fixed instants in the future.

For instance, a future bid is the following: “User X bids for 5 processing power units (i.e. VMs) of type A to be used for 5 hrs, starting at time 13:00, with bid price 0.5 €/min/unit”. Note that in practice the time contains also the Date, but this is omitted for brevity reasons throughout this paper. Note also that different providers may offer the resources required or even a subset thereof; e.g. Provider Y1 offers 2 units from 13:00 to 14:00, while Provider Y2 offers 1 unit from 14:00 to 18:00 and Provider Y3 offers 4 units from 15:00 to 18:00. As opposed to future bids, spot bids demand to utilize resources as soon as they are available. Such bids are distinguished from futures by setting the starting time at a special value (e.g. 0) and by the fact that their start and end time are continuously moving as time passes and the bid is not matched (up to the maximum time allowed by the expiry of the bid). Therefore, spot bids are more flexible than future bids, since they allow users to express demand for service of a certain duration over a larger time interval, as opposed to futures. Note that the actual time of the service of the consumer in this case is a priori unknown, since this depends on when these bids will be matched by asks. For instance, a spot bid is the following: “User X bids for 5 VMs of type A to be used for 5 hrs, starting at time 0, with bid price 0.5 €/min/unit, and time limit 20:00”. In this example the bid could end up be executed the latest starting at 20:00.

3.2 Asks

An ask in our system prescribes the resources offered, which are specified by means of: a) the type and quantity of resources offered, b) the starting time and the end time of the interval when the resources are made available, and c) the total time duration of using the resources. It also specifies d) the price, expressed in €/min/unit and e) the time limit for which the ask is valid and can be used for matching bids. The latter is the maximum time at which the ask is considered to be valid, i.e., the provider of the ask will remove the ask or any remainder of it from the system after the above time. That is, this is the expiry time of the offer, and can be earlier than the maximum time deadline for which the resources offered in the ask can be made available to users.

Similarly with bids, there are also two types of asks, namely future and spot asks. Future asks are those for which the starting and ending times are fixed instants in the future. For asks, the ending time equals the starting time plus the duration, while the time limit also has the same value by default. For instance, a future ask is the following: “Provider Y offers for leasing 2 VMs of type A to be used for 8 hrs, starting at time 15:00, with ask price 0.2 €/min/unit”. On the contrary, spot asks offer resources that can be utilized as soon as there is demand for them. Such asks are distinguished from future asks by setting the starting time at a special value (e.g. 0) and because they are more flexible than future asks, since they offer service of a certain duration over a larger time interval. For instance, a spot ask is the following: “Provider Y offers 2 VMs of type A to be used for 3 hrs, starting at time 0 and until 22:00, with ask price 0.2 €/min/unit, and time limit 19:00”. The semantics is that up to two machines can be used each for up to three time slots (hours), not necessarily consecutive, during the next time slots; this ask will be removed from the system when the time is 19:00 if it has not been matched until then.

3.3 Bid and Ask Queues

Trading is performed by means of a continuous double auction mechanism. This is an extension of the standard spot market mechanism so as to provide for the trading of computation service that can be fully specified only when the associated time is also defined. Similarly to the standard mechanism, the spot bids and asks submitted by traders are placed in the *bid queue* and the *ask queue* respectively. Each queue is ordered according to the price and time of issuance, with the *bid* queue being sorted in *decreasing* order of price, and the *ask* queue being sorted in *increasing* order of price. In the case of the futures market, the bids and asks are listed in a directory service that enables searching and matching.

If two or more orders at the same price appear in a spot queue, then they are entered by time with older orders appearing ahead of newer orders. Since price is discretized in our model, then an equivalent representation of this queue is an ordered (per price) list of queues, one per price asked/bidded, where the asks/bids are sorted by time. The prices displayed to traders when they log into the market are the highest bid price in the bid queue and the lowest ask price in the ask queue. If no price is displayed it is because the corresponding queue is empty.

Orders remain in the queues until they are removed by the system due to expiration, or until they are accepted by other traders (a matching occurred) and result in trades. Expirations are determined according to the terms of the order. In particular, a spot bid expires at its time limit, and the same applies for spot asks.

3.4 Matching

The matching module is invoked when a new bid or ask is submitted. The rationale behind the matching module is that bids are completely satisfied, i.e. there are never remainder bids (parts of a bid that may be satisfied in the future). For simplicity reasons and in order to reduce the potential communication overhead that occurs for customers being served simultaneously by multiple providers, we assume that each bid is served by one provider at any time instance, while multiple providers can only be involved in different times during the servicing of the bid. This assumption will be henceforth referred to as *vertical atomicity*. This assumption is adopted for one more reason: the possibility of serving an application at a certain time with resources belonging to multiple providers depends on the parallelizability of the application, which would then have to be input to the market mechanism and taken into account thereby. Thus, we assume that the matching algorithm considers as candidate matches of a bid only asks whose height is greater than or equal to that of the bid.

Furthermore, we assume that the matching algorithm considers as candidate matches of a bid only asks whose price (per unit) does not exceed that of the bid. Therefore, we omit examining higher price asks and try combine them with lower price asks, even if such combinations could in fact serve the bid with the bidder attaining positive net benefit from the overall charge of the service. This is justified from an economic point of view, since serving customers using higher price asks than the price of the bid would be misleading and distorting for the information signals regarding the actual market price.

Note also that the bid and the ask should be matching in both time and quantity, i.e. all the bid constraints should be satisfied using the existing state of the ask queue. If the bid price and the ask price are different, then the price used is the one of the oldest order. Although this idea is similar to that of a double auction, matters are considerably more complicated because we are dealing with perishable resources with a time dimension. Finally, the matching module must also periodically check for expired bids and asks, which should be removed. In Sect. 4 we present in detail the matching procedure that is to be executed when a new bid/ask is submitted, while in Sect. 5 and Sect. 6 we specify two matching algorithms that could be adopted.

4 Matching and Remainder Asks

The rationale of the matching procedure is to provide the required coverage of the bid with the cheapest matching asks (asks overlapping in time whose price is at most as high as that of the bid) by means of a matching algorithm. If a bid is matched *fully* then reservation of resources, accounting and computation of remainder asks that replace the original asks in the ask queue are performed and the bid is withdrawn from the bid queue and subsequently serviced. Though a bid is always fully matched (fully satisfied), this is not the case for asks. Therefore, in general a fraction of an ask may be used to (partly) match and serve a bid, thus generating a remainder ask.

Specifying the remainders in the futures (forward) market is much simpler than the spot market. Since both future bids and asks are fixed in time, the remainder is a valid ask and can remain in the futures market. This remainder in general corresponds to a non-rectangular shape, in the sense that the amount of VMs offered in not the same for the entire time interval spanned. Such a remainder ask can also be equivalently represented as a collection of at most three rectangular shapes. We henceforth adopt this representation, to clarify the presentation of the algorithms to follow. A related example is depicted in Fig. 3.

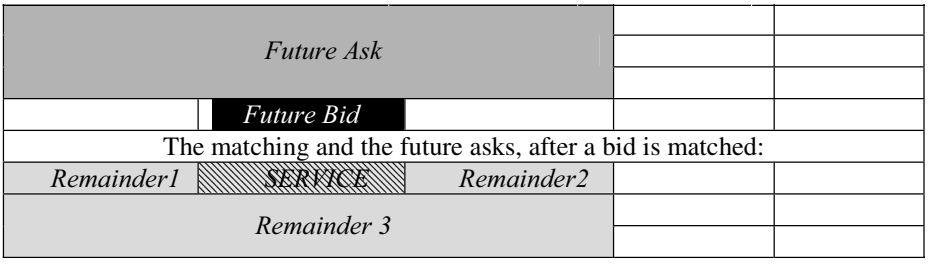


Fig. 3. Matching and remainder asks in the futures market

Things are more complicated for the spot asks, since some of the remainders generated may not be able to offer resource immediately, as opposed to others. The matching procedure and the respective remainders, which are considered as individual rectangles by our matching algorithm, are depicted in Fig. 4 and Fig. 5.

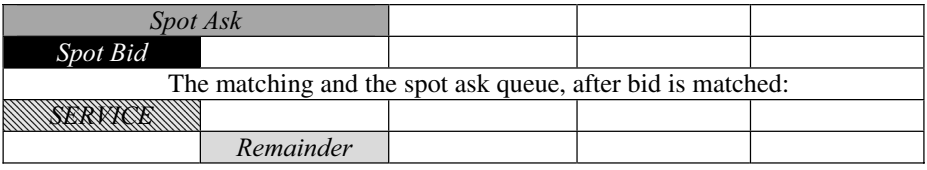


Fig. 4. Matching and remainder of a spot bid with an ask offering same number of VMs

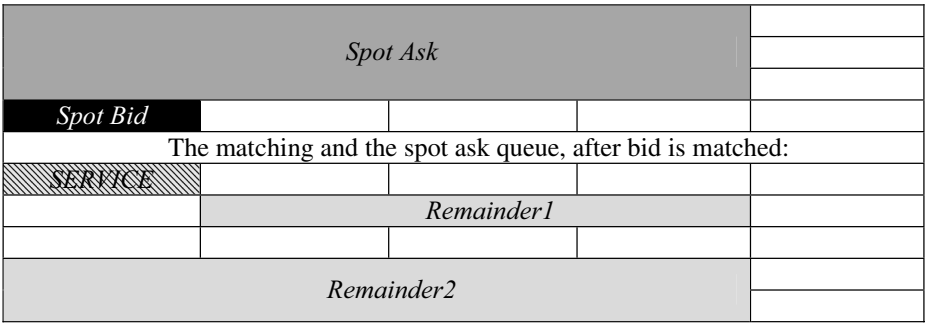


Fig. 5. Matching and remainder of a spot bid with an ask offering different number of VMs

An interesting issue here is that a matching of a spot bid with a much larger spot ask creates remainders that offer resources "as soon as possible" but not immediately, namely *Remainder* of Fig. 4 and *Remainder1* of Fig. 5. There are three options on the treatment of these remainders: a) transfer them to a "waiting queue" and reinsert them into the spot queue when the system time is such that they can indeed offer resources immediately, or b) cancel them and notify the provider, or c) treat all the remainders of a spot ask as valid spot asks which remain in the spot ask queue and are considered for matching, but tagged with additional constraints on when they can be used. Option a) is not economically sound because the market should be kept simple and refrain from making any "brokering" decisions on users' behalf. Indeed, the automated re-insertion of an ask after some time where the market conditions and prices may be completely different than those at the moment, would be confusing for providers who would face uncertainty regarding their strategy. Options b) and c) are both economically sound, with the first being the simplest one, yet resulting in overhead for the hardware providers. On the contrary, option c) does not suffer from this problem, yet it complicates significantly both the representation of asks since now a task description must also include the time slots where the machines of the ask are not free to be used by other bids, and the matching algorithm. In this paper, we investigate both approaches. In particular, option b) is the fundamental assumption for the matching algorithm of Sect. 5. Option c) and its implications on matching are investigated in the penultimate section of this paper, where the outline of a more sophisticated matching algorithm is also presented.

5 The Matching Algorithm

We begin the presentation by focusing on the forward market. For simplicity reasons, it suffices to adopt a one-pass of the queues matching algorithm. The matching algorithm in the futures market is much simpler than that of the spot queue, since the timespan of all bids and asks is fully specified, i.e. their start and end times are decided upon their submission and cannot be changed subsequently, as opposed to spot bids/asks. A meaningful matching procedure for the futures market is to try to match a bid with the cheapest matching asks.

The algorithm for the spot market has to make the same decision but in light of the feature that spot asks may start contributing resources to the matching at some later time, due to the flexibility associated with the provision of their resources. Note that we refrain from adopting a combinatorial approach due to the high computational complexity. The algorithm presented in the remainder of this section is in line with the sorting and treatment of the ask queue in terms of price and time of arrival (in case of equal price for two or more asks), despite the fact that it uses some temporary data structures with a different sorting. Its fundamental property is that if an ask is of lower price than another, then the latter cannot "steal" time of match of the former cheaper ask, i.e. an ask can influence only the quantity of resources that will be provided by higher price asks, as opposed to that of lower asks. This property is very important, since it ensures that the matching algorithm does not violate the rationale of the bid and ask spot market. Also, as mentioned in the previous section, it relies on the assumption that all the spot asks of the queue can offer their resources immediately. This implies that if a spot ask can offer service at some time t, then it can also provide service at any time t' prior to t.

The matching algorithm of this section examines how to cover a particular bid, and produces as the matching solution an ordered list of asks matching the bid in terms of price; the list is ordered with respect to the deadlines of the asks (i.e. latest time to start providing resources). This list would then be passed to the scheduler, who could serve the bid accordingly. However, the algorithm matches the bid with resources taken as much as possible from cheapest asks in the list, which are considered first. That is, the ordering of the list yields the order in time according to which the bid will be served by the various matching asks (or parts thereof).

This code is run from scratch every time a new matching is to be performed, either because a new bid arrived, or because a new ask arrived. Note that when we encounter an ask that could provide some service because its price does not exceed that of the bid we need to decide a) where to place it in the order of asks to serve this bid, b) how much of it to use. The solution we adopt is a) to order the matching asks according to the time constraints b) use as much as possible of cheapest asks. In particular, for any matching ask we use the part of it that does not render any of those asks invalid[1] by any part in time. Indeed if we use less of the specific ask considered than this part, we leave a part of service that could be provided unfulfilled. Yet, if we use a larger portion of it, then we would actually replace service that could be provided by cheaper asks. This would increase the customer's charge and violate the ordering and treatment of asks of the queue with respect to prices. An example of this matching algorithm is provided below:

[1] I.e. it does not cause any time deadline violation due to the "shifting" in time of the service start of the respective ask.

Assume that a spot bid is received, requesting 1 VM for 3 hours for a price of 4€/VM/hr. Assume that there are the following matching asks in the queue, which for simplicity are taken as offering each as many resources as those required by the bid: a) *Ask1*: Offer 1 VM for 1 hour, time deadline: now + 0,5 hour p: 1€/VM/hr. b) *Ask2*: Offer 1 VM for 1 hour, time deadline: now + 1 min p: 2€/VM/hr. c) *Ask3*: Offer 1 VM for 2 hours, time deadline: now + 6 hours p: 3.8€/VM/hr. Note that both for this example and throughout the paper, "now" denotes the start of the next time slot, due to the fact that in our model time is not continuous but discretized in slots.

The algorithm would initially partly match the bid with the **cheapest** ask of the queue, namely *Ask1*. Therefore, the outcome of the execution of the algorithm after examining the first ask in the queue is as follows:

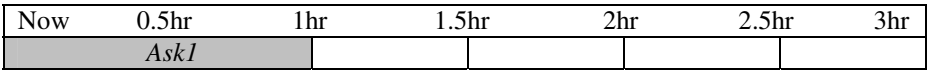


Fig. 6. The algorithm initially partly matches the bid with the cheapest ask

The algorithm would subsequently examine the second cheapest ask, namely *Ask2*. *Ask2* is inserted prior to *Ask1*, due to its shorter deadline. This means that *Ask1* would be shifted in time and then *Ask1* would violate its time constraint by 0.5hr.

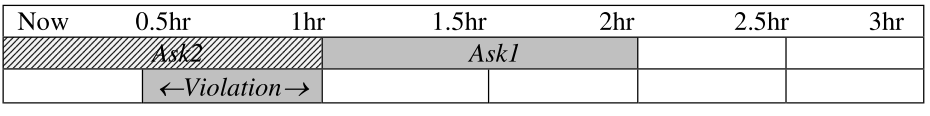


Fig. 7. Ask1 is shifted in time due to the selection of Ask2 as part of the matching solution

This time violation means that only 0.5 hr of service will be provided by *Ask2*, since an ask cannot influence the quantity of resources that will be utilized by any lower price ask, namely *Ask1* in this example. Since we can have in total 1.5 hour of service, our algorithm opts to get as much as possible from the cheapest provider. This means that *Ask2* will provide only 0.5 hr of service, as depicted below:

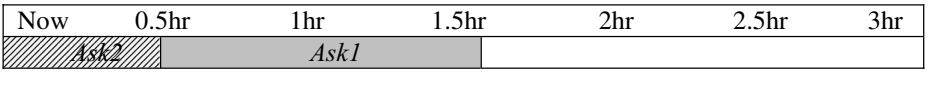


Fig. 8. A fraction of Ask2 is used for the matching since Ask1 must be fully used

Subsequently Ask3 is examined and it provides the remaining 1.5 hr of service. Therefore, this bid will be served as follows:

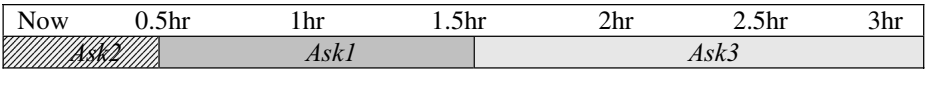


Fig. 9. The solution of the matching algorithm

It is easy to prove that the aforementioned algorithm clearly favors low price asks. By construction, if an ask is of lower price than another, then the latter cannot "steal" time of match of the former cheaper ask, although it can influence its position in the order of providing service to the bids. Therefore, this matching procedure provides nice incentives for providers to submit low price asks. Also, this procedure attempts to match a bid with a low-cost coverage of matching asks.

Yet, this algorithm fails either to always discover matching of a bid with the asks in the queue whenever such a matching is feasible, or guarantee that when it finds a match that this is the lowest-cost match of the bid. This algorithm does not guarantee these properties because its objective is to match the bid completely without violating the queue order. In fact, we have also developed an algorithm that always produces one matching whenever there does exist one. However, the latter is not economically sound because it violates the queue order principle of the bid and ask mechanism and can only serve as a benchmark in order to assess the ratio of matches that the present algorithm misses. To illustrate these shortcomings of the present algorithm, it suffices to modify *Ask2* as follows: Offer 1 VM for 6 hours, time deadline: now + 1 min p: 2€/VM/hr. It is now obvious that the lowest-cost matching for the bid is to match it for its entire duration with *Ask2*; this is depicted as Fig. 10. However, the matching algorithm still returns the same solution, which is depicted as Fig. 11.

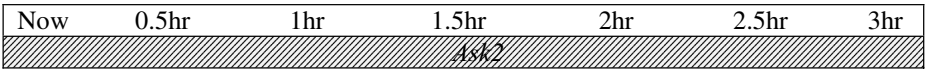


Fig. 10. The cheapest matching solution for the bid submitted



Fig. 11. The matching solution computed by the algorithm

Note that the solution that the matching procedure provides is in fact more expensive, due to the much higher cost of *Ask3*. It is also worth noting that if *Ask3* were absent from the queue, the algorithm would not find the matching with *Ask2* and the bidder would not be served, although this is actually feasible.

Last but not least, we remind the reader that this algorithm relies on the assumption that all spot asks can offer their resources immediately; (remainder) spot asks which could offer resources from some time in the future have been removed from the queue and their providers have been notified accordingly. In addition to the overhead for the hardware providers, the fact that this algorithm works with a subset of the spot asks that could be used for matching bids, further limits the number of matches computed. This is in contrast with the algorithm outlined in the next section, which also favors low price asks and also treats all the remainders of a spot ask as one non-rectangular spot ask which remains in the spot ask queue and is considered for matching.

6 Extensions and Future Work

As opposed to the algorithm of the precious section, we proceed to outline an algorithm, which considers spot asks whose resources are not necessarily available from the current system time (such asks are the *Remainder* of Fig. 4 and *Remainder1* of Fig. 5) as candidate matches. In order to allow such asks in the spot queue, we need to generalize the definition of spot asks (see Sect. 3) so as to be the asks that prescribe that a certain quantity of resources (e.g. 1 VM) for a certain duration (e.g. 2 hours) is made available as soon as possible within certain time intervals (e.g. from 13:00 till 20:00 today, except the intervals [14:00-15:00] and [16:00-17:00] where this VM has already being previously reserved to service some bid) and the ask is valid and present in the queue up to a maximum time deadline, e.g. 18:00 today. Note that this ask is still a spot ask since the starting and ending times are not fixed instants in the future, as opposed to futures.

Note also that this spot ask is different at different times, due to the fact that prior reservations that keep the resources busy are fixed in time. Therefore, the matching of a bid with a set of such asks that are also changing in time is more complicated, in the sense that the algorithm should first specify the current state of the ask. In particular, solving this scheduling problem is a well-known NP-complete problem. Due to the problem's high complexity, we outline an algorithm which is fast enough to be adopted in a realistic market, performs well in terms of the matches computed and does not violate the fundamental rationale of the spot ask queue, i.e. prioritization of cheap asks. Nevertheless, this algorithm is a heuristic approach that does not claim to solve the scheduling problem, i.e. it cannot always compute a set of matching asks for a bid if there is indeed one. Its formal definition and assessment are beyond the scope of this paper and are left for future work. However, the rationale of the algorithm is presented below.

The algorithm initially computes the candidate matches for the ask (i.e. asks of the demanded quantity of VMs) that can offer service from time *Now* (denoting the start of the next time slot) and for a service duration equal to that specified in the bid. This is performed by means of creating a matrix. Such an example matrix is depicted as Fig. 12. Each column of the matrix corresponds to a slot of the time interval where service will be provided. Each row corresponds to a provider that can offer service within this time interval, with the cheapest being on the top row. The cells where each provider can offer service are marked, as well as the total availability of each provider's (i.e. number of slots where they can offer the desired amount of VMs). For instance, Provider2 in Fig. 12 can offer three hours of computation anywhere within the 4-hour time interval, i.e. provider's availability is 3. If there is a slot where it is not possible to provide service for any provider, then the algorithm fails and proceeds to find a match for the time interval [Now + 1 slot, Now + 1 slot + service duration]. The algorithm then detects the slots where there is only one provider offering service; these providers are matched for those slots and their total availability for service is subsequently reduced. Then the algorithm attempts to fill the slots where there are multiple candidate providers, regardless of their total availability: For these slots, the algorithm attempts to do a probabilistic matching. In particular, the algorithm starts with the cheapest ask and according to the provider's availability randomly fills some slots, so that the provider's availability becomes zero. I.e. the cheapest ask is fully

Now	+1hr	+2hr	+3hr	+4hr	Availability
Unavailable	Provider1	Provider1	Provider1		2
Provider2	Provider2	Provider2	Provider2		3
Provider3	Provider3	Unavailable	Provider3		3

Fig. 12. The matrix used from Algorithm 2 to match a spot bid demanding 4 hours of service

utilized. It then proceeds with the next cheapest ask and does the same. Note however that after the second step, there might be slots allocated to two candidate providers. For these slots, each provider is assigned a probability of moving from this slot, depending on his total availability. A dice is thrown and a provider is moved to an empty slot according to a transition probability, which is larger for slots where the number of providers that could serve this slot is small. The algorithm terminates when all the slots are assigned to some provider and thus a match is found. In case there are slots where there is no provider serving it, while there are not any slots with more than one provider, the algorithm has failed to compute a match. Due to the matching algorithm's probabilistic nature, it can be repeated for a maximum prespecified number of times until it computes a match. If it fails, then it attempts to compute a match at a next time window, i.e. at the second time for the time interval [Now + 1 slot, Now + 1 slot + service duration]. This is performed until a match is indeed found or the algorithm fails for the entire duration where the bid is valid.

Note that for some services, e.g. non-parallel distributed applications, such as a company's web server, it might be meaningful to enforce *horizontal atomicity* instead of *vertical atomicity*. This means that the user should be assigned a provider's VM for the entire duration of service demanded. However, multiple providers may offer the total number of VMs requested. If this is indeed the case, the matching algorithm is greatly simplified. The reason is that under this assumption, candidate asks are only the asks of providers that can offer VMs for the entire duration of service demanded by the bid. Therefore, the algorithm sorts the asks providing a VM for the entire duration within each time interval, starting from [Now, Now + service duration]. If the number of matching asks in this interval is at least that demanded, then the cheapest VMs are selected and provided as match. If the number of matching asks is less, there is no possible match and the algorithm proceeds to compute a match at a next time window, i.e. at the second time for the time interval [Now + 1 slot, Now + 1 slot + service duration]. This is performed until a match is indeed found or the algorithm fails for the entire duration where the bid is valid. It is trivial to prove that this algorithm never fails to detect a match if any and that it also always computes the cheapest matching ask that can be provided as soon as possible to the user.

Throughout the paper we have assumed that customers are interested in rate of computation in a certain time interval. Replacing the "rectangles" of this market with a total quantity of computation greatly simplifies the matching algorithms presented earlier applicable for this market as well. Thus, instead of trying to match a bid with a rectangle constructed by a set of asks with proper height, the matching algorithm simply picks the cheapest asks that can provide the desired computation within the specified deadline.

As already mentioned it is possible that a bid be satisfied by the asks submitted by multiple providers. This clearly increases the switching costs of users and reduces the value of the allocations of the market. Therefore, this problem should be mitigated by means of a special algorithm. Such an algorithm could prescribe that units allocated to different users should be "swapped" if possible, thus resulting in a less fragmented with respect to number of providers per user, outcome. It is worth emphasizing that though units of allocation can be swapped between consumers, prices and quantities are not. A preliminary idea for such an algorithm is to swap units between two users, if and only if for some performance index (e.g. total number of different asks matched) the post-swap value is better for one user while being non-worse for the other. The formal definition of such an algorithm, as well as conducting simulations for the evaluation of the algorithms presented in this paper, is left for future work.

7 Conclusions

In this paper, we have specified a market where hardware providers can interact with users interested in leasing Grid resources for a price and a time period. Our approach comprises a stock-market like mechanism that enables the trading of computational power on the basis of a spot and a futures market. The spot market comprises a pair of bid and ask queues. This grid market is more complicated than the standard spot/futures markets of storable commodities, because the computational service traded in our case comprises of resources that are perishable, and has both quantity and duration specified in terms of a time interval. This is an important feature of our market mechanism that has been taken into account by both the market mechanism and the related matching algorithms that operate on the spot bid/ask queues and futures market. Finally, we have briefly addressed the issue of post-sale optimization in order to mitigate the switching cost of consumers being served by multiple providers over time. The formal definition of such an algorithm is left for future work. Another direction for future research is to formally specify and evaluate the algorithm that provides matchings according to which service can start with a delay, due to the fact that remainder asks that do not provide readily available resources are employed.

Acknowledgments. This work has been supported in part by the European Commission within the Framework Programm FP7, ICT, through the project GridEcon. The authors would like to thank all consortium members for useful discussions on the subject of this paper.

References

1. Buyya, R., Abramson, D., Venugopal, S.: The Grid Economy. Proceedings of the IEEE 93(3), 698–714 (2005)
2. The Gridecon Project Consortium: Deliverable D2.1. Economic Modelling Requirement Report (2008), http://www.gridecon.eu/html/deliverables.shtml
3. Buyya, R., Abramson, D., Giddy, J., Stockinger, H.: Economic Models for Resource Management and Scheduling in Grid Computing. The Journal of Concurrency and Computation Practice and Experience (CCPE) (2002)

4. Denton, M.J., Rassenti, S.J., Smith, V.L.: Spot market mechanism design and competitivity issues in electric power. In: Proceedings of the Thirty First HICS
5. Lien, D., Quirk, J.: Measuring the Benefits from Futures Markets: Conceptual Issues. International Journal of Business and Economics 1(1), 53–58 (2002)
6. http://www.amazon.com/gp/browse.html?node=201590011
7. http://www.sun.com/service/sungrid/index.jsp
8. http://h20338.www2.hp.com/enterprise/cache/250417-0-0-225-121.html

Mapping heavy communication Workflows onto Grid Resources within an SLA context

Dang Minh Quan

Paderborn Center of Parallel Computing, University of Paderborn, Germany

Abstract. Service Level Agreements (SLAs) are currently one of the major research topics in Grid Computing. Among many system components for supporting SLA-aware Grid jobs, the SLA mapping mechanism receives an important position. It is responsible for assigning sub-jobs of the workflow to Grid resources in a way that meets the user's deadline and as cheap as possible. With the distinguished workload and resource characteristics, mapping a heavy communication workflow within SLA context defines new problem and needs new method to be solved. This paper presents the mapping algorithm, which can cope with the problem. Performance measurements deliver evaluation results on the quality and efficiency of the method.

1 Introduction

Mapping and running jobs on suitable resources are the core tasks in Grid Computing. With the case of Grid-based workflows, where a single job is divided into several sub-jobs, the majority of efforts for this issue concentrate on finding a mapping solution in best effort manner[1–3]. In the SLA (Service Level Agreement) context, where resources are reserved to ensure the Quality of Service (QoS), mapping a workflow requires different mechanism. The literature recorded some proposed solutions for this problem in [4–6]. Most of the proposed mechanisms suppose a workflow including many sub-jobs, which are sequent programs, and a Grid service having ability to handle one sub-job at a time. This is not sufficient enough as sub-jobs in many existed workflows [7–9] are parallel programs, and many High Performance Computing Centers (HPCCs) provide computing service under single Grid service [10]. It is obvious that a HPCC can handle many sub-jobs, which can be either sequent programs or parallel programs, at a time. Moreover, all of them did not consider the case of having heavy communication among sub-jobs in the workflow. This paper, which is a continuous work in a series of efforts supporting SLA for the Grid-based workflow [12–14], will present a mechanism to handle all stated drawbacks.

1.1 Workflow model

Like many popular systems handling Grid-based workflow [1–3], we also suppose Directed Acyclic Graph (DAG) form of the workflow. User describes the

specification about the required resources to run sub-jobs, data transfer among sub-jobs, the estimated runtime of sub-jobs and impose the expected runtime of the whole workflow. User wants the system to finish running the whole workflow in time. In the scope of this paper, the time is computed in slot. Each slot equals with a specific period of real time. Figure 1 presents a concrete Grid workflow. Each sub-job of the workflow has different resource requirements as described in table 1.

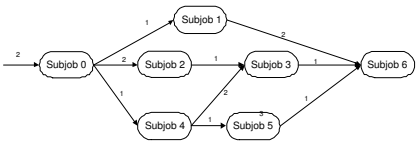


Fig. 1. A sample workflow

Sj_ID	CPU	Storage	exp	runtime
0	18	59	1	6
1	16	130	3	8
2	20	142	4	5
3	22	113	4	8
4	18	174	2	9
5	20	97	3	14
6	16	118	1	4

Table 1. Resource requirements for sub-jobs

It is noted that a sub-job of the workflow can be either a single program or a parallel program and the data to be transferred among sub-jobs is very large, usually in GB scale. The case of light communication among sub-jobs of the workflow was handled in [14]

1.2 Grid service model

The computational Grid includes many High Performance Computing Centers (HPCCs). We believe that only HPCCs have enough conditions to support SLA for sub-jobs of the workflow. The resources in each HPCC are managed by software called local Resource Management System (RMS). In this paper, the acronym RMS is used to represent the HPCC as well as the Grid service of that HPCC. Each RMS has its own resource configuration and this configuration is usually different from other RMSs. Those differences include number of CPU, number of memory, storage capacity, software, expert, service price, etc. To ensure that the sub-job can be executed within a dedicated time period, the RMS must support advance resource reservation, for example CCS [10]. Figure 2 depicts a sample CPU reservation profile in such RMS. Queuing-based RMSs are not suitable for our requirement, as no information about the starting time is provided. In our system, we reserve three main types of resource: CPUs, storages and experts. An extension to other devices is straightforward.

If two sequent sub-jobs are executed in the same RMS, it is not necessary to do data transfer task and the time used for this task equal to 0. Otherwise, the data transfer task must be performed. To make sure that a specific amount of data will be transferred within a specific period of time, the bandwidth must also

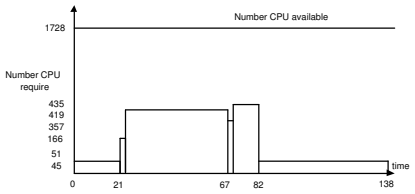


Fig. 2. A sample CPU reservation profile of a local RMS

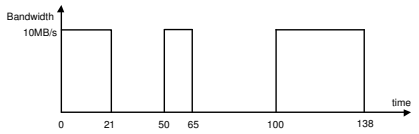


Fig. 3. A sample bandwidth reservation profile of a link between two local RMSs

be reserved. Unfortunately, up to now, there is no mechanism responsible for that task in the worldwide network. Here, to overcome that elimination, we use central broker mechanism. The link bandwidth between two local RMSs is determined as the average bandwidth between two sites in the network. Whenever having a data transfer task on a link, the SLA broker will determine which time slot is available for that task. During that specified period, the task can use the whole bandwidth and other tasks must wait. Using this principal, the bandwidth reservation profile of a link will look similar to the one as depicted in Figure 3. A more correctly model with bandwidth estimation [11] can be used to determine the bandwidth within a specific time period instead of the average value. In both cases, the main mechanism is unchanged.

1.3 Mapping mechanism requirement

The formal specification of the described problem includes following elements:

- Let R be the set of Grid RMSs. This set includes a finite number of RMSs, which provide static information about controlled resources and the current reservations/assignments.
- Let S be the set of sub-jobs in a given workflow including all sub-jobs with the current resource and deadline requirements.
- Let E be the set of data transfer in the workflow, which express the dependency between the sub-jobs and the necessity for data transfers between the sub-jobs.
- Let K_i be the set of resource candidates of sub-job s_i. This set includes all RMSs, which can run sub-job s_i, $K_i \subset R$.

Based on the given input, a feasible and possibly optimal solution is sought, which allows the most efficient mapping of the workflow in a Grid environment with respect to the given global deadline. The required solution is a set defined as

$$M = \{(s_i, r_j, start_slot) | s_i \in S, r_j \in K_i\} \tag{1}$$

A feasible solution must satisfy following conditions:

- The total runtime period of the workflow must be within the expected period given by user.

- All $K_i \neq \emptyset$. There is at least one RMS in the candidate set of each sub-job.
- The dependencies of the sub-jobs are resolved and the execution order remains unchanged.
- Each RMS provides a profile of currently available resources and can run many sub-jobs of a single flow both sequentially and parallel. Those sub-jobs, which run on the same RMS, form a profile of resource requirement. With each RMS r_j running sub-jobs of the Grid workflow, with each time slot in the profile of available resources and profile of resource requirements, the number of available resources must be larger than the resource requirement.

In the next phase the feasible solution with the lowest cost is sought. The cost of a Grid workflow is defined as a sum of four factors: money for using CPU, money for using storage, cost of using experts knowledge and finally money for transferring data between the involved resources. If two sequent subjobs run on the same RMS, the cost of transferring data from the previous subjob to the later subjob is neglected. It can be shown easily that the optimal mapping of the workflow to Grid RMS with cost optimizing is a NP hard problem.

2 Related work

In two separated works [5, 6], Zeng et al and Iwona et al built systems to support QoS features for Grid-based workflow. In their work, a workflow includes many sub-jobs, which are sequent programs, and a Grid service has ability to handle one sub-job at a time. To map the workflow on to the Grid services, they used Integer Programming method. Applying Integer Programming to our problem faces many difficulties. The first is the flexibility in runtime of the data transfer task. The time to complete data transfer task depends on the bandwidth and the reservation profile of the link, which varies from link to link. The variety in completion time of data transfer task makes the constraints presentation very complicated. The second is that a RMS can handle many parallel programs at a time. Thus, presenting the constraints of profile resource requirement and profile of resource available in Integer Programming is very difficult to perform.

With the same resource reservation and workflow model, we proposed an algorithm which mapping a light communication workflow to Grid resources in [14]. The proposed algorithm uses Tabu search to find the best possible assignment of sub-jobs to resources. In order to shorten the computation time caused by the high number of resource profiles to be analyzed and by the flexibility while determining start and end times for the sub-jobs, several techniques for reducing the search space are introduced. However, these techniques cannot be applied to solve the problem in this paper because of different workload context.

Metaheuristics such as GA, Simulated Annealing [15], etc were proved to be very effective in mapping, scheduling problems. McGough et al also use them in their system [4]. However, in our problem, with the appearance of resource profiles, the evaluation at each step of the search is very hard. If the problem is big with highly flexible variable, the classical searching algorithms need very

long time to find a good solution. In the scope of this paper, we apply several standard Metaheuristics to our problem as means of comparing.

3 Planning algorithm for heavy communication workflows

The input of the mapping procedure includes information about workflow and information about RMSs. Information about workflow is provided in a file describing sub-jobs and a file describing the dependence. Information about RMSs is stored in a relational database. They include the description of the resource configuration in each RMS, the resource reservation profile of each RMS and the bandwidth reservation profile of each link. The information is collected from RMSs by the monitoring module. Based on this information, the system will do mapping. The overall mapping mechanism, which is called H-Map, is presented in Figure 4.

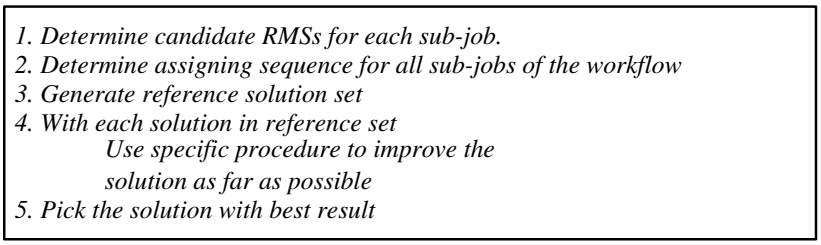


Fig. 4. Mapping mechanism overview

3.1 Determining candidate RMSs for each sub-job

Each sub-job has different resource requirement about type of RMS, type of CPU, etc. There are a lot of RMSs with different resource configuration. This phase finds among those heterogeneous RMSs the suitable RMSs, which can meet the requirement of each sub-job. Each resource parameter of an RMS is represented by number value and is stored in a separate column in the database table. For example, with the parameter Operating System, Linux, Sun, Window, Unix are represented by value number 1, 2, 3, 4 respectively. The co-relative resource requirement parameter of a sub-job is also represented by number value in the same manner. Thus, the matching between sub-job's resource requirement and RMS's resource configuration is done by several logic checking conditions in the WHERE clause of the SQL SELECT command.

3.2 Determining the assigning sequence of the workflow

When the RMS to execute each sub-job, the bandwidth among sub-jobs was determined, the next task is determining time slot to run sub-job in the specified

RMS. At this point, the assigning sequence of the workflow becomes important. The sequence of determining runtime for sub-jobs of the workflow in RMS can also affect the total runtime especially in the case of having many sub-jobs in the same RMS.

In general, to ensure the integrity of the workflow, sub-jobs in the workflow must be assigned basing on the sequence of the data processing. However, that principle does not cover the case of a set of sub-jobs, which have the same priority in data sequence and do not depend on each other. To examine the problem, we determine the earliest and the latest start time of each sub-jobs of the workflow in ideal condition. The time period to do data transfer among sub-jobs is computed by dividing the amount of data to a fix bandwidth. The earliest and latest start, stop time for each sub-job and data transfer depends only to the workflow topology and the runtime of sub-jobs but not the resources context. Those parameters can be determined by using conventional graph algorithms. A sample of those data for the workflow in Figure 1, in which the number above each link represents number of time slots to do data transfer, is presented in Table 2.

```
assign_number of each candidate RMS =0
While m_size < max_size {
  clear similar set
  foreach sub-job in the workflow {
    foreach RMS in the candidate list {
      foreach solution in similar set {
          if solution contains sub-job:RMS
            num_sim++
          store tuple (sub-job, RMS, num_sim) in
          a list }}
    sort the list
    pick the best result
    assign_number++
    If assign_number > 1
      find defined solution having the same
      sub-job:RMS and put to similar set
}}
```

Fig. 5. The algorithm generate reference set

Sub-job	Earliest start	Latest start
0	0	0
1	7	22
2	8	17
3	18	23
4	7	7
5	17	17
6	32	32

Table 2. Valid start time for sub-jobs of workflow in Figure 1

The ability of finding a suitable resource slot to run a sub-job depends on number of resource free during the valid running period. From the graph, we can see sub-job 1 and sub-job 4 having the same priority in data sequence. However, from the data in table 2, sub-job 1 can start at max time slot 22 while sub-job 4 can start at max time slot 7 without affecting the finished time of workflow. Suppose that two sub-jobs are mapped to run in the same RMS and the RMS can run one sub-job at a time. If sub-job 4 is assigned first at time slot 7, sub-job 1 will be run from time slot 16 thus the workflow will not be late. If

sub-job 1 is assigned first, in the worse case at time slot 22, sub-job 4 can be run at time slot 30 and the workflow will late 23 time slots. Here we can see, the latest time factor is the main parameter to evaluate the full affection of the sequence assignment decision. It can be seen through the affection, mapping the sub-job having smaller latest start time first will make the latency smaller. Thus, the latest start time value determined as above can be used to determine the assigning sequence. The sub-job having smaller latest start time will be assigned earlier.

3.3 Generating reference solution set

A solution is found by determining each sub-job of the workflow run by which RMS. We do not consider time factor in this phase so a reference solution is defined as a set of map sub-job:RMS with all sub-jobs in the workflow. Each solution in the reference solutions set can be thought as the starting point for local search so it should be spread as wide as possible in the searching space. To satisfy the space spreading requirement, number of the same map sub-job:RMS between two solutions must be as small as possible. The number of member in the reference set depends on the number of available RMSs and number of sub-jobs. During the process of generating reference solution set, each candidate RMS of a sub-job has a co-relative $assign_number$ to count the times that RMS is assigned to the sub-job. During the process of building a reference solution, we use a similar set to store all defined solution having at least a map sub-job:RMS similar to one in the creating solution. The algorithm is defined in Figure 5.

While building a solution, with each sub-job in the workflow, we select the RMS in the set of candidate RMSs, which creates minimal number of similar sub-job:RMS with other solutions in the similar set. After that, we increase the $assign_number$ of the selected RMS. If this value larger than 1, which means that the RMS were assigned to the sub-job more than one time, there must exist solutions that contains the same sub-job:RMS and thus satisfying the similar condition. We search those solutions in the reference set, which have not been in the similar set, and then add them to similar set. When finished, the solution is put to the reference set. After all reference solutions are defined, we use a specific procedure to refine each of the solution as far as possible.

3.4 Improving solution quality algorithm

Before improving the quality of the solution, we have to determine specific run-time period for each sub-job and each data transfer task as well as the makespan of the present solution. The start time of a data transfer task depends on the finish time of the source sub-job and the state of the link's reservation profile. We use min_st_tran variable to present the dependence on the finish time of the source sub-job. The start time of a sub-job depends on the latest finish time of the related data transfer tasks and the state of the RMS's reservation profile. We use min_sj_tran variable to present the dependence on the latest finish time of

the related data transfer tasks. The task to determine timetable for the workflow is done with the procedure in Figure 6.

```
foreach sub-job k following the assign sequence {
  foreach link from determined sub-jobs to k{
    min_st_tran=end_time of source sub-job
    search reservation profile of link the
    start_tran > min_st_tran
    end_tran = start_tran+num_data/bandwidth
    store end_tran in a list
  }
  min_st_sj=max (end_tran)
  search in reservation profile of RMS running
  k the start_job > min_st_sj
  end_job= start_job + runtime
}
```

Fig. 6. Algorithm determine timetable for workflow

```
while (num_loop < max_loop) {
 foreach subjob in the workflow {
   foreach RMS in the candidate list {
     if cheaper then put (sjid, RMS id, improve_value)
     to a list }}
 sort the list according to improve_value
 from the begin of the list{
   Compute time table to get the finished time
   If finished time < limit
     break
 }
 Store the result
 num_loop ++;
}
```

Fig. 7. Procedure to improve the solution quality

For each sub-job of the workflow in the assigning sequence, firstly, we find all the runtime period of data transfer task from previous sub-jobs to current sub-job. This period must be later than the finish time of the source sub-job. Note that with each different link the transfer time is different because of different bandwidth. Then, we determine the runtime period of the sub-job itself. This period must be later than the latest finish time of previous related data transfer task. The whole procedure is not so complicate but time consuming. The time consuming steps are the searching reservation profiles, and they make the whole procedure long time consuming.

The overall of solution quality improvement procedure is described in Figure 7. In one iteration, we can move only one sub-job to one RMS with the hope to decrease the cost. So we only consider the move, which can decrease the cost. With each solution we compute the time table, if it satisfies the deadline then update the result.

4 Performance evaluation

Performance experiment is done with simulation to check for the quality of the mapping algorithms. The hardware and software used in the experiments is rather standard and simple (Pentium 4 2,8Ghz, 2GB RAM, Linux Redhat 9.0, MySQL). The whole simulation program is implemented in C/C++. The goal of the experiment is to measure the feasibility, the quality of the solution and the time needed for the computation. To do the experiment, 18 workflows with different topologies, number of sub-jobs, sub-job specifications, amount of data transferring were generated and mapped to 20 RMSs with different resource configuration and different resource reservation context by 6 algorithms H-Map, Simulated Annealing (SA), Guided Local Search (GLS), Iterated Local Search (ILS), Genetic Algorithm (GA), Estimation of Distribution Algorithm (EDA)

[15]. The implementation of those algorithms is described in [16]. The final result of the experiment is presented in table 3 with column Wf (Workflow) presents the id of workflows, column Rt (Runtime) and Cost record the cost and runtime of solutions generated by each algorithm correlative with each workflow respectively.

	H-Map		SA		ILS		GLS		GA		EDA	
Wf	Rt	Cost	Rt	Cost	Rt	Cost	Rt	Cost	Rt	Cost	Rt	Cost
1	1	632.58	105	632.58	21	632.58	14	632.58	25	632.58	62	632.58
2	0.5	756.96	84	756.90	37	756.96	19	756.90	22	756.96	88	756.90
3	1	780.42	114	780.42	54	780.42	29	780.34	27	780.34	126	780.34
4	1	836.18	78	836.18	81	836.18	46	836.18	28	836.18	178	836.18
5	1	892.02	105	892.02	114	892.02	59	892.02	29	892.21	241	892.02
6	2	948.10	90	948.10	147	948.10	80	948.10	36	1005.27	390	947.86
7	1	1003.7	78	1003.99	201	1003.7	98	1003.99	36	1075.19	462	1003.7
8	2	1059.89	121	1059.89	250	1059.89	127	1059.89	32	1059.89	558	1059.89
9	2	1184.21	130	1184.21	307	1184.21	167	1184.21	44	1248.86	659	1183.92
10	2	1308.53	146	1332.53	398	1308.53	187	1308.53	47	1383.53	680	1308.53
11	3	1364.14	124	1376.42	502	1364.14	222	1377.63	52	1440.81	956	1364.42
12	2	1488.12	184	1551.74	462	1521.74	303	1501.95	51	1569.39	854	1536.74
13	7	1512.26	174	1512.26	620	1512.26	354	1566.09	56	1620.17	1136	1512.26
14	3	1567.74	162	1631.15	815	1567.74	392	1568.15	56	1663.81	1255	1601.15
15	6	1591.12	161	1675.67	876	1591.12	524	1621.67	70	1764.18	1663	1621.67
16	5	1786.56	180	1871.81	1394	1840.31	763	1843.55	85	1914.91	2845	1830.87
17	7	1889.78	197	1960.87	1695	1892.27	1258	1936.83	93	2028.06	4170	1961.30
18	10	2217.34	272	2276.33	2046	2283.67	1623	2256.53	1953	2406.97	10976	2276.33

Table 3. Experiment results of the H-Map algorithm

The experiment results show that H-Map algorithm finds out higher quality solution with much shorter runtime than other algorithms in most cases. Some of the metaheuristics such as ILS, GLS, EDA find out equal results with small problems. But with big problem, they have exponent runtime and find out unsatisfied results.

5 Conclusion

This paper has presented a method, which performs an efficient and precise assignment of heavy communication workflow to Grid resources with respect to SLAs defined deadlines and cost optimization. In our work, the distinguished character is that a sub-job of the workflow can be a sequent or parallel program and a Grid service can handle many sub-jobs at a time. The performance evaluation showed that the proposed algorithm creates solution of equal or better quality than most standard metaheuristics and needs significantly shorter computation time. The latter is a decisive factor for the applicability of the method

in real environments, because large-scale workflows can be planned and assigned efficiently.

References

1. E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S. Patil, M. Su, K. Vahi, and M. Livny, "Pegasus : Mapping Scientific Workflows onto the Grid", Proceedings of the 2nd European Across Grids Conference, Nicosia, Cyprus, January 28-30, 2004.
2. D. P. Spooner, S. A. Jarvis, J. Cao, S. Saini and G. R. Nudd, "Local Grid Scheduling Techniques Using Performance Prediction", IEEE Proceedings - Computers and Digital Techniques, 150(2), pp. 87–96, 2003.
3. R. Lovas, G. Dzsa, P. Kacsuk, N. Podhorszki, D. Drtos, "Workflow Support for Complex Grid Applications: Integrated and Portal Solutions", Proceedings of 2nd European Across Grids Conference, Nicosia, Cyprus, 2004.
4. S. McGough, A. Afzal, J. Darlington, N. Furmento, A. Mayer, and L. Young ,Making the Grid Predictable through Reservations and Performance Modelling, The Computer Journal, v.48 n.3, pp. 358–368, 2005
5. L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, H. Chang, QoS-Aware Middleware for Web Services Composition, IEEE Transactions on Software Engineering, v.30 n.5, pp. 311–327, may 2004
6. I. Brandic and S. Benkner and G. Engelbrecht and R. Schmidt, QoS Support for Time-Critical Grid Workflow Applications, Proceedings of e-Science 2005
7. S. Ludtke, P. Baldwin, and W. Chiu, "EMAN: Semiautomated Software for High-Resolution Single-Particle Reconstructio" , Journal of Structure Biology, v. 128, 1999.
8. G. B. Berriman, J. C. Good, A. C. Laity, "Montage: a Grid Enabled Image Mosaic Service for the National Virtual Observatory" , ADASS, v. 13, 2003.
9. L. Richter, "Workflow Support for Complex Grid Applications: Integrated and Portal Solutions", Proceedings of the 2nd European Across Grids Conference, 2004.
10. L. Burchard, M. Hovestadt, O. Kao, A. Keller, and B. Linnert, "The Virtual Resource Manager: An Architecture for SLA-aware Resource Management", Proceedings of the IEEE CCGrid 2004, IEEE Press, pp. 126–133, 2004.
11. R. Wolski, "Experiences with Predicting Resource Performance On-line in Computational Grid Settings", ACM SIGMETRICS Performance Evaluation Review, v. 30 n. 4, pp. 41–49, 2003.
12. D.M. Quan, O. Kao, "SLA negotiation protocol for Grid-based workflows", Proceedings of the International Conference on High Performance Computing and Communications (HPPC-05), LNCS 3726, pp. 505–510, 2005.
13. D.M. Quan, O. Kao, "On Architecture for an SLA-aware Job Flows in Grid Environments", Proceedings of the 19th IEEE International Conference on Advanced Information Networking and Applications (AINA 2005) , IEEE Press , pp. 287–292, 2005.
14. D.M. Quan, O. Kao, "Mapping Grid job flows to Grid resources within SLA context", Proceedings of the European Grid Conference,(EGC 2005), LNCS 3470, pp. 1107–1116, 2005.
15. C. Blum, A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison", ACM Computing Surveys, v. 35 n.3, pp. 268–308, 2003
16. D.M. Quan, "A Framework for SLA-aware execution of Grid-based workflows", PhD thesis, University of Paderborn - Germany, 2006.

Mapping a group of jobs in the error recovery of the Grid-based workflow within SLA context

Dang Minh Quan
International University in Germany
School of Information Technology
Bruchsal 76646, Germany
quandm@upb.de

Jorn Altmann
International University in Germany
School of Information Technology
Bruchsal 76646, Germany
jorn.altmann@acm.org

Abstract

The error recovery mechanism receives an important position in the system supporting Service Level Agreements (SLAs) for the Grid-based workflow. If one sub-job of the workflow is late, a group of directly affected sub-jobs should be re-mapped in a way that does not affect the start time of other sub-jobs in the workflow and is as inexpensive as possible. With the distinguished workload and resource characteristics as well as the goal of the problem, this problem needs new method to be solved. This paper presents a mapping algorithm, which can cope with the problem. Performance measurements deliver good evaluation results on the quality and efficiency of the method.

1. Introduction

Service Level Agreements (SLAs) [13] are currently one of the major research topics in Grid Computing, as they serve as a foundation for a reliable and predictable job execution at remote Grid sites. We have developed the system supporting SLA for the Grid-based workflow [8, 10, 9]. The scenario of running a Grid-based workflow is presented in Figure 1.

Like many popular systems handling Grid-based workflows [3, 14, 6], our proposed workflow system is of the Directed Acyclic Graph (DAG) form. It is noted that a sub-job of the workflow can be either a sequential program or a parallel program and that the data to be transferred among sub-jobs can be enormous.

In each Grid site, the resources are managed by the software called local Resource Management System (RMS). Each RMS has its own unique resource configuration. To ensure that the sub-job can be executed within a dedicated time period, the RMS must support advance resource reservation such as CCS [5].

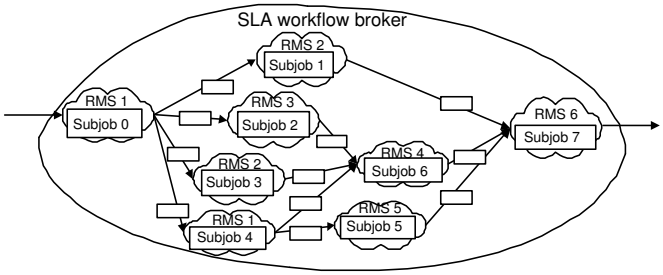


Figure 1. A sample running Grid-based workflow scenario

If two sequential sub-jobs are executed in the same RMS, it is not necessary to do data transfer and the time used for this work equals zero. Otherwise, data transfer tasks between the RMS's must be performed. To make sure that a specific amount of data will be transferred within a specific period of time, a bandwidth must also be reserved. The mechanism to reserve the bandwidth is described in [11, 12].

During the sub-job running process, an error inside an RMS may happen at any time and can damage the whole workflow. The error can be node scratch, hardware failure, network cable break. In this case, the RMS will restart the sub-job from the checkpointing image. The time overhead to detect the error and the time to rerun the sub-job from the checkpointing image will make the finished time of the sub-job later than the pre-determined deadline. When one sub-job is late, its output data transfer to the next sub-jobs cannot be done. Therefore, those next sub-jobs cannot run in the right way because of lacking input data and the whole wokflow is damaged. For example, if sub-job 0 is late 1 time slot, sub-job 1, 2, 3 and 4 are also affected because of having no input data.

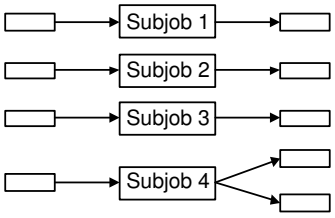


Figure 2. Affected sub-jobs and data transfers

Therefore, we have to try to re-map the directly affected sub-jobs in a way that does not affect the start time of other remaining sub-jobs in the workflow. When we re-map the directly affected sub-jobs, we also have to re-map their related data transfers. With the example in Figure 1, if sub-job 0 is late, the affected sub-jobs and data transfers are described in Figure 2. This task can be feasible because of many reasons.

- The latency period is very small, only 1 or 2 time slots.
- The Grid may have others solutions that the data transfers will be shorter because the links have broader bandwidth.
- The Grid may have RMSs with higher CPU power, which can execute the sub-jobs in shorter time.

The formal specification of the described problem includes following elements:

- Let R be the set of Grid RMSs. This set includes a finite number of RMSs, which provide static information about controlled resources and the current reservations/assignments.
- Let S be the set of sub-jobs in the given workflow.
- Let Sa be the set of all directly affected sub-jobs with the resource and runtime requirements.
- Let E be the set of data transfer in the given workflow.
- Let Ei be the set of input data transfers of S.
- Let Eo be the set of input data transfers of S.
- Let K_i be the set of resource candidates of sub-job s_i. This set includes all RMSs, which can run sub-job s_i, $K_i \subset R$, $s_i \in Sa$.

Based on the given input, a feasible and possibly optimal solution is sought, which allows the most efficient mapping of those sub-jobs in a Grid environment with respect to the given deadlines. The required solution is a set defined as Formula 1.

$$M = \{(s_i, r_j, start_slot) | s_i \in Sa, r_j \in K_i\} \quad (1)$$

If the solution does not have *start_slot* for each s_i, it become a configuration as defined in Formula 2.

$$a = \{(s_i, r_j | s_i \in Sa, r_j \in K_i\} \quad (2)$$

A feasible solution must satisfy following conditions:

- **Criteria1:** All $K_i \neq \emptyset$. There is at least one RMS in the candidate set of each sub-job.
- **Criteria2:** The start time of each input data transfer ei_h must later than the sub-job it depends on, $ei_h \in Ei$. The stop time of each output data transfer eo_k must earlier than the next sub-job which depends on it, $eo_k \in Eo$.
- **Criteria3:** Each RMS provides a profile of currently available resources and can run many sub-jobs both sequentially and parallel. Those sub-jobs, which run on the same RMS, form a profile of resource requirement. With each RMS r_j running sub-jobs of Sa, with each time slot in the profile of available resources and profile of resource requirements, the number of available resources must be larger than the resource requirement.
- **Criteria4:** The data transmission task e_{ki} from sub-job s_k to sub-job s_i must not overlap other reserved data transmission task on the link between RMS running sub-job s_k to RMS running sub-job s_i, $e_{ki} \in Ei \cup Eo$, $s_k, s_i \in Sa$.

In the next phase the feasible solution with the lowest cost is sought. The cost C of a Grid workflow is defined in formula 3, 4 and 5. It is a sum of four factors: the cost of (1) using the CPU, (2) storage, (3) expert knowledge, and (4) data transfer between resources.

$$C_1 = \sum_{i=1}^{n} s_i.r_t * (s_i.n_c * r_j.p_c + s_i.n_s * r_j.p_s + s_i.n_e * r_j.p_e) \quad (3)$$

$$C_2 = \sum e_{ki}.n_d * r_j.p_d \quad (4)$$

$$C = C_1 + C_2 \quad (5)$$

with $s_i.r_t, s_i.n_c, s_i.n_s, s_i.n_e$ are the runtime, number CPU, number storage, number expert of sub-job s_i respectively. $r_j.p_c, r_j.p_s, r_j.p_e, r_j.p_d$ are the price of using CPU, storage, expert, data transmission of RMS r_j respectively. $e_{ki}.n_d$ is the number of data to be transferred from sub-job s_k to sub-job s_i.

If two sequential sub-jobs run on the same RMS, the cost of transferring data from the previous sub-job

to the later sub-job is neglected. It can be shown easily that the optimal mapping of those sub-jobs to Grid RMSs with cost optimizing is a NP hard problem.

This paper presents a mapping algorithm called G-map to handle this problem. It is the next progress in a series of efforts, [8, 10, 9, 7, 11, 12], to build a full system supporting SLAs for Grid-based workflows.

2. Related work

Our problem can be defined as a special case of the mapping a bag of independent tasks to resources problem [15, 4, 1]. In the literature, most of the efforts [15, 4, 1] solve this problem with time optimization. It is the major difference to our work because we concentrate to find out a solution which meets the deadline and optimizes the cost.

The most closed work to our problem is the work from [2]. In [2], the authors present the method to schedule parameter sweep applications on global Grids. The original deadline and budget constrained (DBC) costtime optimization algorithm builds on the cost-optimization and time-optimization scheduling algorithms. This is accomplished by applying the time-optimization algorithm to schedule task-farming application jobs on distributed resources having the same processing cost. The authors assume that all tasks are sequential programs and identical to each other. It is clear that our problem is distinguished from parameter sweep applications scheduling problem as the sub-jobs in our problem are parallel programs with various differences in configurations as well as in constraints. However, the primary idea of DBC algorithm can also be applied to our problem as presented in Figure 3.

```
For each sub-job in the set {
   Sort the candidate RMSs according to cost order. If 2 or
    more RMSs have the same cost, the more powerful RMS
    will stay first
    For each RMS in the sorted candidate list {
        calculate the execution time of that sub-job on the
         RMS. If it meet the deadline then assigned the sub-
         job to the RMS
}}
```

Figure 3. Application of DBC algorithm to the problem

In [12], we present an algorithm called H-Map to map heavy communication workflow to the Grid resources. The character of resources and workload are similar to this problem. Therefore, we can easily adapt H-Map to the problem as described in Figure 4. The main idea of H-Map algorithm is that a set of initial configuration distributed over the search space according to cost factor will be further refined to find the best solution.

```
Create set of reference configurations distributed over the
search space according to cost factor
for each sub-job in the set {
    for each RMS in the candidate list {
        if cheaper then put (sjid, RMS id, improve_value)
        to a list }}
sort the list according to improve_value
from the begin of the list{
    Compute time table to get the finished time
    If finished time < limit
        break
}
Store the result
```

Figure 4. Application of H-Map algorithm to our problem

As the number of affected sub-jobs is not always big we can apply the searching all cases algorithm (SAC). In the case of having small number of affected sub-jobs, the runtime of the algorithm is sufferable. The solution result of this algorithm can be a good reference source to evaluate the quality of other algorithms. The searching all cases algorithm is presented in Figure 5.

```
min_cost=1000000
With each possible configuration a {
    determine the scheduling order
    check the feasibility of a
    if feasible {
        compute the cost m_cost of a
        if(m_cost<min_cost){
            min_cost=m_cost
            store a
} } }
```

Figure 5. Searching all cases algorithm

3. G-map algorithm

G-Map algorithm maps a group of sub-jobs on to the Grid resources with G stands for Group. In the G-Map algorithm, we try to compress the solution space in a way that the ability to have feasible solutions is higher. After that, a set of initial configuration is constructed. This set will be improved by local search until it cannot be improved any more. Finally, we pick the best solution from the final set. The architecture of the algorithm is presented in Figure 6.

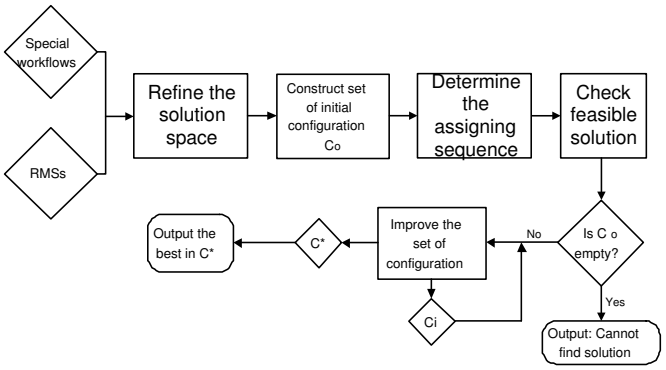


Figure 6. G-Map algorithm

3.1. Refining the solution space

The set of candidate RMSs for each sub-job can be continuously refined by following observation: An RMS will be valid with a sub-job only if the sub-job assigned to that RMS satisfies the start time of the next sequential sub-jobs. The algorithm to refine the solution space is presented in Figure 7.

With each separate sub-job, we determine the schedule time of the input data transfers, the sub-job and output data transfer. From the algorithm in Figure 7, we can see that the resource reservation profile is not updated. We call this the ideal assignment. If the stop time of the output data transfer does not earlier than the start time of the next sequential sub-job then we remove the RMS out of the candidate set.

3.2. Constructing the set of initial configurations

The goal of the algorithm is finding out a feasible solution, which satisfies all required criteria and is as inexpensive as possible. Therefore, the set of initial configurations should satisfy two criteria.

```
for each sub-job k in the set {
  for each RMS r in the candidate list of k{
    for each link to k in assigned sequence{
      min_st_tran=end_time of source sub-job
      search reservation profile of link  the
      start_tran > min_st_tran
      end_tran = start_tran+num_data/bandwidth
      update reservation profile
    }
    min_st_sj=max (end_tran)
    search in reservation profile of  r the
    start_job > min_st_sj
    end_job= start_job + runtime
    for each link from k in assigned sequence{
      min_st_tran=end_job
      search reservation profile of link  the
      start_tran > min_st_tran
      end_tran = start_tran+num_data/bandwidth
      update reservation profile
      if end_tran>=end_time of destination sub-job
        remove r       out of the candidate list
}}}
```

Figure 7. Refining the solution space procedure

- The configurations in the set must differ from each other as far as possible. This criterion will ensure that the set of initial configuration will distribute widely over the search space.
- The RMS running sub-job in each configuration should differ from each other. This criterion will ensure that each sub-job will be assigned in ideal condition, thus the ability to become a feasible solution will increase.

The procedure to create the set of initial configuration is as following.

Step1: Sorting the candidate set according to the cost factor. With each sub-job, we compute the cost of running the sub-job by each RMS in the candidate set and then sort the RMSs according to the cost. With the case of our example, we could have a sorted solution space as presented in Figure 8.

Step2: Forming the first configuration. The procedure to form the first configuration in the set is presented in Figure 9. We form the first solution having as small cost as possible. With each unassigned sub-job, we compute the m_delta = cost running in the first feasible RMS minus cost running in the second feasible RMS in the sorted candidate list. The sub-job having the smallest m_delta will be assigned to the first feasible RMS. The purpose of this action is to en-

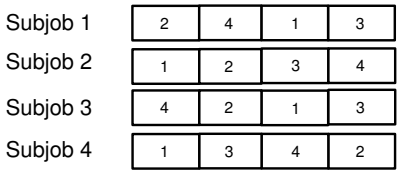


Figure 8. A sample sorted solution space

sure that the sub-job having higher ability to increase the cost will be assigned first. After that, we will update the reservation profile and check if the assigned RMS is still available for other sub-jobs. If not we will mark it as unavailable. The process is repeated until all sub-jobs are assigned. The selection of which sub-job to be assigned is effective when there are many sub-jobs having the same RMS as the first feasible solution.

```
While the set of unassigned sub-jobs is not empty {
   Foreach sub-job s in the set of unassigned sub-jobs {
      m_delta=cost in first feasible RMS- cost in second
      feasible RMS
      put (s, RMS, m_delta) in a list
   }
   Sort the list to get the minimum m_delta
   Assign s to the RMS
   Drop s out of the set of unassigned sub-jobs
   Update the reservation profile of the RMS
   Check if the RMS is still feasible with other unassigned
   sub-jobs
   if not, mark the RMS is infeasible
}
```

Figure 9. The algorithm to form the first configuration

Step3: Forming the other configurations. The procedure to form other initial configurations is described in Figure 10. To satisfy two criteria as described above, we use $assign_number$ to keep track of number of the assignment RMS to a sub-job and l_ass to keep track of the appearance frequency of RMS within a configuration. RMS having smaller $assign_number$ and small appearance frequency in l_ass will be selected. Applying this algorithm to the example, starting from the first configuration in Step 2, the process of forming other initial configurations is described in Figure 11.

```
assign_number of each candidate RMS =0
While number of configuration < max_sol {
  clear list of assigned RMS l_ass
  for each sub-job in the set {
     find in the candidate list RMS r having the
     smallest number of appearance in l_ass
     and the smallest assign_number
     Put r to l_ass
     assign_number++
}}
```

Figure 10. Procedure to create the initial configuration set

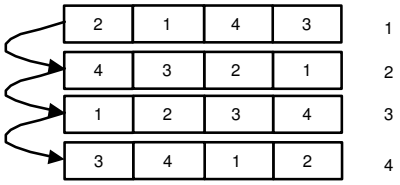


Figure 11. The sample initial configuration set

3.3. Determining the assigning order

When the RMS executing each sub-job and the bandwidths among sub-jobs were determined, the next task is determining time slot to run a sub-job in the specific RMS. At this time, the order of determining scheduled time for sub-jobs becomes important. The sequence of determining runtime for sub-jobs in RMS can also affect the Criteria 2 especially in the case of having many sub-jobs in the same RMS. In this algorithm, we use the policy like in [12]. Thus, the input data transfer having the earliest start time smaller will be scheduled earlier. The output data transfer having the latest stop time smaller will be scheduled earlier. The sub-job having earlier deadline should be scheduled earlier.

3.4. Checking the feasibility of a solution

To check the feasibility of a solution we have to determine the timetable to execute sub-jobs and their related data transfer. In the error recovery phase, finding a solution that meet the Criteria 2 is very important. Therefore, we do not simply use the provided runtime of each sub-job but modify it according to the performance of each RMS. Let pk_i, pk_j is the performance of a CPU in RMS r_i, r_j respectively and $pk_j > pk_i$. Suppose that a sub-job has the provided runtime rt_i with the resource requirement equals to r_i. Thus, the run-

time rt_j of the sub-job in r_j is determined as in Formula 6.

$$rt_j = \frac{rt_i}{\frac{pk_i+(pk_j-pk_i)*k}{pk_i}} \quad (6)$$

Parameter k presents the affection of the sub-job's communication character and the RMS's communication infrastructure. For example, if pk_j equals to $2*pk_i$ and rt_i is 10 hours, rt_j will be 5 hours if k equals to 1. However, k=1 only when there are no communication among parallel tasks of the sub-job. Otherwise, k will be less than 1. Parameter k_a is an average value, which is determined by the user through many experiments and is provided as the input for the algorithm. In the reality environment, k may fluctuate around the average value depending on the network infrastructure of the system. For example, suppose that k_a equals to 0.8. If the cluster has good network communication, the real value of k may increase to 0.9. If the cluster has not so good network communication, the real value of k may decrease to 0.7. Nowadays, with the very good network technology in High Performance Computing Centers, the fluctuation of k is not so much. To overcome the fluctuation problem , we use the pessimistic value k_p instead of k in the Formula 6 to determine the new runtime of the sub-job as following.

- If $k_a > 0.8$, for example with the rare communication sub-job, $k_p = 0.5$.
- If $0.8 > k_a > 0.5$, for example with normal communication sub-job, $k_p = 0.25$.
- If $k_a < 0.5$, for example with heavy communication sub-job, $k_p = 0$.

The pessimistic policy will ensure that the sub-job can be finished within the new determined runtime period. With those assumption, the procedure to determine the timetable is presented in Figure 12.

After determining the timetable, the stop time of the output data transfer will be compared with the start time of the next sequential sub-jobs. If having a violation, this solution is determined infeasible.

3.5. Improving solution quality algorithm

If the initial configuration set $C_0 \neq \emptyset$, the set will gradually be refined to have better quality solutions. The refining process stops when the solutions in the set cannot be improved any more and we have the final set $C*$. The best solution in $C*$ will be output as the result of the algorithm. More detail about this procedure can be found in [12].

```
for each sub-job k in the set {
  for each link to k in assigned sequence{
     min_st_tran=end_time of source sub-job
     search reservation profile of link  the
     start_tran > min_st_tran
     end_tran = start_tran+num_data/bandwidth
     update link reservation profile
  }
  min_st_sj=max (end_tran)
  search in reservation profile of RMS running
  k the start_job > min_st_sj
  end_job= start_job + runtime
  update resource reservation profile
  for each link from k in assigned sequence{
    min_st_tran=end_job
    search reservation profile of link  the
    start_tran > min_st_tran
    end_tran = start_tran+num_data/bandwidth
    update link reservation profile
}}
```

Figure 12. Procedure to determine the timetable

4. Performance experiment

Performance experiment is done with simulation to check for the quality of the G-Map algorithm. The goal of the experiment is comparing the quality of G-Map algorithm with other algorithms in different workloads and resource contexts. The quality of an algorithm is evaluated by several factors: the ability of finding feasible solution, the cost of the found solution, the execution time. The hardware and software used in the experiments are rather standard and simple (Pentium D 2,8Ghz, 1GB RAM, Fedora Core 5, MySQL). The total simulation program includes about 5000 lines of C/C++ code.

To compare the quality of all described algorithms above, we generated 8 different workflows which have different topologies, different sub-job specifications, different amount of data transfers, different the maximum number of the potential directly affected sub-jobs, from 1 to 10.

As the difference in the static factors of an RMS such as OS, CPU speed and so on can be easily filter by SQL query, we use 20 RMSs with the resource configuration equal or even better than the requirement of sub-jobs. Those RMSs have already had some initial workload in their resource reservation profiles and bandwidth reservation profiles. Those 8 workflows are mapped to 20 RMSs. We select the late sub-job in each workflow in a way that the number of the directly af-

fected sub-jobs equals the maximum number of the potential directly affected sub-jobs of that workflow. The late period is 1 time slot. With each group of the affected sub-jobs, we change the power configuration of RMS and the k value of affected sub-jobs. Those configurations are presented in Table 1. For example, with the first row in the Table 1, the resource configuration 90-0-10 means that there is 90% number of RMS having CPU performance like requirement, 0% number of RMS having CPU performance 25% more power than requirement, 10% number of RMS having CPU performance 50% more power than requirement. The workload configuration 90-0-10 means that 90% number of affected sub-jobs having $k = 0.5$, 0% number of affected sub-jobs having $k = 0.25$, 10% number of affected sub-jobs having $k = 0$.

Resource configuration			Workload configuration		
0%	25%	50%	k=0.5	k=0.25	k=0
90	0	10	90	0	10
90	10	0	90	10	0
60	30	10	60	30	10
60	10	30	60	10	30
30	60	10	30	60	10
30	10	60	30	10	60
10	30	60	10	30	60
10	60	30	10	60	30
10	0	90	10	0	90
10	90	0	10	90	0
0	10	90	0	10	90
0	90	10	0	90	10

Table 1. Resource configuration scenario and workload configuration scenario

With each affected sub-job group, with each power resource configuration scenario, with each workload configuration scenario, we do mapping with 4 algorithms: G-Map, DBC, H-Map, search all cases. Thus, with each algorithm, we have total 8*12*12=1152 running instances. With each running instance we record the runtime of the algorithm and the cost of the solution if it is feasible.

Table 2 presents the detail cost and runtime of each algorithm for different group of affected sub-jobs in three extreme experimental scenario. Among 4 algorithms, only the SAC algorithm have great runtime when the size of the problem increase. The runtime of this algorithm becomes exponent when number of affected sub-job greater than or equal 6. Other algorithms have very small runtime. In all cases, the runtime of H-Map, DBC, G-Map is not greater than 1 second.

	SAC		H-Map		DBC		G-Map	
Sjs	Cost	Rt	Cost	Rt	Cost	Rt	Cost	Rt
Resource configuration 90-10-0 and Workload configuration 0-10-90								
3	50.3	4	50.3	0.5	50.3	0.5	50.3	0.5
4	62.0	59	62.0	1	62.0	1	62.0	1
5	87.7	1211	87.7	1	87.7	1	87.7	1
6	-	-	78.4	1	81.5	1	78.5	1
7	-	-	95.1	1	102.3	1	95.1	1
8	-	-	113.5	1	113.5	1	113.5	1
9	-	-	110.7	1	119.7	1	110.7	1
10	-	-	144.6	1	150.6	1	142.8	1
Resource configuration 60-30-0 and Workload configuration 30-60-0								
3	50.3	4	50.3	0.5	50.3	0.5	50.3	0.5
4	N/S	60	N/S	1	N/S	1	N/S	1
5	87.7	1217	87.7	0.5	87.7	1	87.7	1
6	-	-	81.5	1	92	0.5	81.5	1
7	-	-	95.1	1	102.3	1	95.1	0.5
8	-	-	N/S	0.5	113.5	0.5	113.5	1
9	-	-	N/S	1	N/S	1	N/S	0.5
10	-	-	N/S	0.5	166.2	1	153.6	1
Resource configuration 0-10-90 and Workload configuration 90-10-0								
3	50.3	4	50.3	0.5	50.3	0.5	50.3	0.5
4	N/S	57	N/S	1	N/S	1	N/S	1
5	87.7	1205	87.7	0.5	87.7	1	87.7	1
6	-	-	81.5	1	92	0.5	81.5	1
7	-	-	95.1	1	102.3	1	95.1	1
8	-	-	N/S	0.5	N/S	0.5	N/S	1
9	-	-	N/S	1	N/S	1	N/S	1
10	-	-	N/S	0.5	166.2	1	153.6	1

Table 2. Performance result in three extreme experimental scenario

From the data in the table, we can see clearly a trend that if the Grid has a lot of more powerful RMSs than requirement and the group of affected sub-jobs has a lot of computing intensive jobs (big k), the chance to have a feasible solution will be higher and vice versa. Because of having the greatest runtime, the search all cases algorithm can find out solution within an acceptable period when the size of the problem is small. Therefore, it has the smallest ability to find a feasible solution. H-Map algorithm also has limited ability to find a feasible solution. The reason is that H-Map is designed for mapping the whole workflow but not the special case like this problem. Thus, there are a lot of infeasible solutions in the initial configuration set. G-Map and DBC algorithm have the same ability to find

a feasible solution. Figure 13 presents the total number of finding out feasible solution in the whole experiment co-relative with each algorithm.

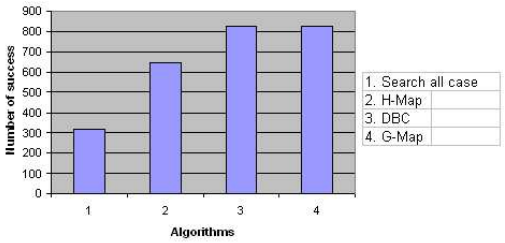


Figure 13. Total number of feasible found solution by algorithms

From the data of the experiment, we can see the domination of local search approach to the quality of the solution. If H-Map or G-Map algorithm can find a feasible solution, it is high quality solution with low cost. As search all case and H-Map have smaller ability to find a feasible solution. We only compare the quality of the solution between G_Map and DBC algorithms. The experimental data shows that G-Map find lower cost solution than DBC. The average cost in relative value between G-Map and DBC is 1 versus 1.05.

5. Conclusion

This paper has presented a method, which performs an efficient and precise assignment of a group of sub-jobs in the error recovery of the Grid-based workflow within SLA context with respect to ensure the start time of other sub-jobs and cost optimization. In our work, the distinguished character is that those sub-jobs of the workflow can be a sequential or parallel program and a Grid service can handle many sub-jobs at a time. The performance evaluation showed that the proposed algorithm creates solution of equal or better quality than most existed applicable algorithms within a very short time period. Short execution time and high quality solution are the decisive factor for the applicability of the method in real environments, because the error recovery can be performed efficiently.

References

[1] T. Braun et al. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, 61(6):810 – 837, 2001.

[2] R. Buyya et al. Scheduling parameter sweep applications on global grids: A deadline and budget constrained cost-time optimisation algorithm. *Software: Practice and Experience (SPE), Wiley Press*, 35(5):491 – 512, 2005.

[3] E. Deelman. Mapping abstract complex workflows onto grid environments. *Journal of Grid Computing*, 1(1):25–39, 2003.

[4] Y. Gao et al. Adaptive grid job scheduling with genetic algorithms. *Future Gener. Comput. Syst.*, 21(1):151 – 161, 2005.

[5] M. Hovestadt. Scheduling in hpc resource management systems:queuing vs. planning. *Proc. 9th Workshop on JSSPP at GGF8, LNCS*, pages 1–20, 2003.

[6] R. Lovas et al. Support for complex grid applications: Integrated and portal solutions. *Proc. 2nd European Across Grids Conference*, May 2004.

[7] D. Quan et al. Mapping grid job flows to grid resources within sla context. *Proceedings of the European Grid Conference,(EGC 2005)*, 3470:1107–1116, 2005.

[8] D. Quan et al. On architecture for an sla-aware job flows in grid environments. *Journal of Interconnection Networks, World scientific computing*, pages 245–264, 2005.

[9] D. Quan et al. On architecture for an sla-aware job flows in grid environments. *Proceedings of the 19th IEEE International Conference on Advanced Information Networking and Applications (AINA 2005) , IEEE Press*, pages 287–292, 2005.

[10] D. Quan et al. Sla negotiation protocol for grid-based workflows. *Proceedings of the International Conference on High Performance Computing and Communications (HPPC-05), LNCS 3726*, pages 505–510, 2005.

[11] D. Quan et al. Error recovery mechanism for grid-based workflow within sla context. *Accepted by International Journal of High Performance Computing and Networking (IJHPCN)*, 2006.

[12] D. Quan et al. Mapping heavy communication workflows onto grid resources within sla context. *Proceedings of the International Conference of High Performance Computing and Communication (HPCC06)*, 2006.

[13] A. Sahai. Automated sla monitoring for web services. *DSOM 2002, LNCS*, 2506:28–41, 2002.

[14] D. P. Spooner et al. Local grid scheduling techniques using performance prediction. *IEEE Proc. Computers and Digital Techniques*, pages 87–96, May 2003.

[15] A. Sulistio et al. A time optimization algorithm for scheduling bag-of-task applications in auction-based proportional share systems. *Proceedings of the 17th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2005, IEEE CS Press)*, 2005.

Mapping of SLA-based workflows with light communication onto Grid resources

Dang Minh Quan[1] and Jörn Altmann[1 2]

[1] International University of Bruchsal, Campus 3, 76646 Bruchsal, Germany
dang.minh@i-u.de
[2] TEMEP, School of Engineering, Seoul National University, South-Korea
jorn.altmann@acm.org

Abstract. Service Level Agreements (SLAs) are currently one of the major research topics in Grid Computing. Among those system components that support SLA-aware Grid jobs, the SLA mapping mechanism has an important position. It is responsible for assigning sub-jobs of the workflow to Grid resources in a way that meets the user's deadline and minimizes costs. Assuming many different kinds of sub-jobs and resources, the process of mapping an SLA-based workflow with light communication defines an unfamiliar and difficult problem. This paper presents a solution to this problem. The quality and efficiency of the algorithm is validated through performance measurements.

1 Introduction

Service Level Agreements (SLAs) define a business contract between users and service providers. SLAs in the context of Grid computing describe an business environment, in which providers guarantee users that their workflow will be executed on the Grid within the agreed upon period of time and will get paid for the service usage. One of the main system components, which support SLAs management for Grid-based workflows is the job mapping mechanism. The mapping mechanism that we will present maps each sub-job of the workflow to resources in a manner that satisfies two main criteria: First, it will guarantee the workflow execution on time; Second, it will calculate a low cost solution.

If a customer uses a service, he is charged based on the service usage and can expect a quality as indicated in the SLA. An automated mapping is necessary as it frees users from the tedious job of assigning sub-jobs to resources under many constraints such as workflow integrity and execution deadline, cost minimisation. Additionally, a good mapping mechanism will help users to minimize the cost for using Grid resources.

We have developed a running system for SLA-based workflows [7–10], which differs from other works [5, 6] in the following aspects. In our system, sub-jobs of the workflow can be sequential or parallel programs and each resource can manage many sub-jobs at the same time. Our work on this topic solved the problem for workflows with heavy communication. The key difference of this paper and the previous works is the communication. The light communication can help us

ignore the complex in time and cost caused by data transfer. Thus, we could apply specific technique to improve the speed and the quality of the mapping algorithm. An initial solution for the case of workflows with light communication has been presented in [7], but without considering variable runtime of sub-jobs and different resource requirements.

This paper, which belongs to this series of efforts to develop a job mapping framework for SLA-based workflows [7–10], will present a solution for variable runtime of sub-jobs and different resource requirements.

Like many popular systems handling SLA-based workflows [1–3], our workflow system is of the Directed Acyclic Graph (DAG) form. It is assumed that the data to be transferred among sub-jobs is very small, usually less than 10MB. The user is also required to specify the estimated runtime of each sub-job together with the specific resource requirements. The time is split into slots. Each slot represents a constant period of real time, in the range of 2 to 5 minutes. Figure 1 illustrates an example of a Grid workflow with resource requirements as specified in Table 1. The label of each link of the workflow represents the amount of data, which has to be transferred between two sub-jobs.

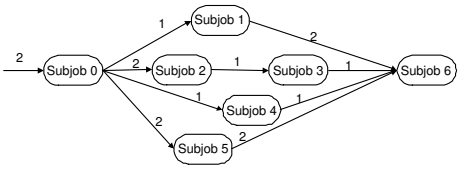


Fig. 1. A sample workflow

Sj_ID	CPU	Storage	Exp	Runtime
0	64	59	1	18
1	48	130	2	16
2	64	142	1	17
3	64	113	2	20
4	48	174	1	18
5	48	97	1	19
6	64	118	1	18

Table 1. Resource requirements for sub-jobs

At each Grid site, which is assumed to be a High Performance Computing Center (HPCC), the resources are managed by the software system that is called local Resource Management System (RMS). Each RMS has its own unique resource configuration. To ensure that a sub-job can be executed within a specified time period, the RMS must support advance resource reservation such as CCS [12]. In our system, RMS has the capability of reserving three main types of resources: CPUs, storages, and experts. An extension to further resources is straightforward. To illustrate the working of the solution by means of an example, we assume to have three RMSs, which have the same number of CPUs, namely 96, the same amount of storage, 3TB, and the same number of experts, 10. All reservation profiles are empty.

HPCCs are usually inter-connected by a broadband link, which is greater than 100Mbps. The length of one time slot in our system is between 2 and 5 minutes. Thus, the amount of data transferred through a link within one time slot can range from 1.2GB to 3GB. Since we assume less than 10MB of data transfer

between sub-jobs (workflows with light communications), the data transfer can easily be performed within one time slot (right after the sub-job had finished its calculation) without affecting any other communication between two RMSs.

A formal specification of the described problem entails the following elements:

- Let R be the set of all Grid RMSs. It includes a finite number of RMSs, which provide static information about controlled resources and the current reservations/allocations.
- Let S be the set of all sub-jobs in a given workflow. It includes all sub-jobs with the current resource and deadline requirements.
- Let E be the set of all data transfers in a workflow. It indicates the dependency between the sub-jobs and the necessity for data transfers between the sub-jobs.
- Let K_i be the set of resource candidates for sub-job s_i. It includes all RMSs that can run sub-job s_i. We note that K_i is a subset of R.

Based on the given input, we are looking for a feasible and, possibly, cost-minimizing solution. The required solution is a set defined in Formula 1.

$$M = \{(s_i, r_j, start_slot) | s_i \in S, r_j \in K_i\} \tag{1}$$

If the solution does not have start_slot for each s_i, it becomes a configuration as defined in Formula 2.

$$a = \{(s_i, r_j | s_i \in S, r_j \in K_i\} \tag{2}$$

A feasible solution must satisfy the following conditions:

- **Criteria1:** All $K_i \neq \emptyset$. There is at least one RMS in the candidate set of each sub-job.
- **Criteria2:** The total runtime period of the workflow must be within the expected period given by the user.
- **Criteria3:** The dependency among the sub-jobs is resolved and the execution order remains unchanged.
- **Criteria4:** Each RMS provides a profile of currently available resources and can run many sub-jobs of a single workflow in parallel. Those sub-jobs, which run on the same RMS, form a profile of resource requirements. For each RMS r_j with its profile of available resources and for each time slot, , the number of available resources must be larger than the resource requirements.
- **Criteria5:** The data transmission task e_{ki} from sub-job s_k to sub-job s_i takes one timeslot right after the finished time of sub-job s_k , $e_{ki} \in E$. If sub-job s_k and sub-job s_i are executed in the same RMS the data transfer task is neglected. This can be assumed since all compute nodes in a cluster usually use a shared storage system like NFS or DFS. Thus, the time to move data in the same storage is zero.

In the next phase, feasible solutions with the lowest cost are sought. As the number of data to be transferred between sub-jobs in the workflow is very small,

we can omit the cost of data transfer. Thus, the cost C of a Grid workflow is defined in Formula 3. It is the sum of the charge of using: (1) the CPU, (2) the storage and (3) the expert knowledge.

$$C = \sum_{i=1}^{n} s_i.r_t * (s_i.n_c * r_j.p_c + s_i.n_s * r_j.p_s + s_i.n_e * r_j.p_e) \quad (3)$$

with $s_i.n_c, s_i.n_s, s_i.n_e$ being the number of CPUs, the amount of storage, the number of expert required for sub-job s_i respectively. $s_i.r_{tj}$ is the runtime of sub-job s_i in RMS r_j. The value of $s_i.r_{tj}$ can be determined with the mechanism described in [13]. $r_j.p_c, r_j.p_s, r_j.p_e$ are the price of using CPU, storage, expert of RMS r_j respectively.

It can easily be shown that the cost-minimizing mapping of a workflow onto RMSs is a NP-hard problem.

2 Related works

In [5, 6], Zeng et al and Iwona et al built systems to support QoS features for SLA-based workflows. To map a workflow onto a Grid resource, they used Integer Programming (IP) for finding a solution. Applying IP to our problem is impossible because of two reasons. First, the arbitrary length of a sub-job cannot be expressed in an Integer Programming model. The time to complete a sub-job depends on the resource configuration and the reservation profile of the RMS. Second, an RMS can handle many parallel programs at the same time. Thus, we do not know how many, which, and when sub-jobs will be executed in an RMS. Consequently, we cannot formulate the IP constraint of available resource as described in Criteria 4.

Meta-heuristics such as Genetic Algorithm and Simulated Annealing proved to be very effective in mapping and scheduling problems. However, in the case of our problem, with the appearance of resource profiles, the evaluation at each step of the search becomes expensive for large problems [10].

A mechanism (based on Tabu search) for mapping a light communication workflow onto Grid resources is described in [7]. In order to shorten the computation time caused by the high number of resource profiles to be analyzed and by the wider range of start time of the sub-job without violating the end time, several techniques for reducing the search space were introduced. However, these techniques cannot be applied to solve the problem in this paper because of the variable runtime lengths of the sub-jobs within different RMSs.

The work in [10] solved the problem of mapping a heavy communication workflow onto Grid resources. The H-Map algorithm has been proposed. The main idea of the H-Map algorithm is forming a widely distributed set of initial configurations and performing the local search for each of them. As the problem in [10] is closed to the problem in this paper, we can adapt the H-Map algorithm to map the light communication workflow onto Grid resources. The framework is retained as described in Figure 2, but we change the computing timetable function and computing cost function to suit the new requirement.

```
Sort the solution space according to the computation cost
Clear the initial set of solutions
While not enough solutions {
   Form new configuration by combining 2 cost levels
   Compute timetable to check the feasible
   If feasible, put to the initial set of solutions
}
For each solution in the set {
   Do local search with the cost function
}
Pick the best solution
```

Fig. 2. Framework of the H-Map algorithm

```
Analyze the workflow into set of sub-jobs in sequential
layers. Sub-jobs in the same layer do not depend on each
other.
For each sub-job in the set {
   Sort the candidate RMSs according to cost order.
   For each RMS in the sorted candidate list {
      calculate the execution time of that sub-job on the
      RMS. If it meet the deadline then assigned the sub-
      job to the RMS
}}
```

Fig. 3. The application of DBC algorithm to our problem

The original DBC Grid scheduling algorithm [11], called the cost-time optimization scheduling algorithm, is used to schedule parameter sweep application on global Grids. This algorithm builds on the cost-optimization and time-optimization scheduling algorithms. This is accomplished by applying the time-optimization algorithm for scheduling task-farming jobs onto distributed resources having the same processing cost. Even this algorithm only supports sequential sub-jobs. However, the idea can also be applied to our problem since our workflow can be considered a parameter sweep application. The modified algorithm is presented in Figure 3.

3 L-Map algorithm

In this paper, we propose an algorithm called L-Map to map light communication workflows onto the Grid RMSs (L - stands for light). The goal of the L-Map algorithm is to find a solution, which satisfies the requirements as described in Section 1.

Each sub-job has different resource requirements. There are a lot of RMSs with different resource configurations. The initial action is to find the suitable RMSs among those heterogeneous RMSs, which can meet the requirement of the sub-job. The matching between the sub-job's resource requirements and the RMS's resource configuration is done by several logic checking conditions in the WHERE clause of the SQL SELECT command. This work will satisfy Criteria 1. Suppose that each sub-job has m RMSs in the candidate list, we could have m^n configurations. The overall L-Map algorithm is presented in Figure 4. The following sections will describe each procedure of the algorithm in detail.

Step 0: With each sub-job s_i, we sort the RMSs in the candidate set K_i according to the cost of running s_i. The cost is computed according to Formula 3. The configuration space of the sample is presented in Table 2. Each RMS-Rt column presents the RMS and the runtime of the sub-job in this RMS.

Step 1: We form the first configuration by assigning each sub-job to the RMS having the lowest cost in the candidate list. The calculated configuration

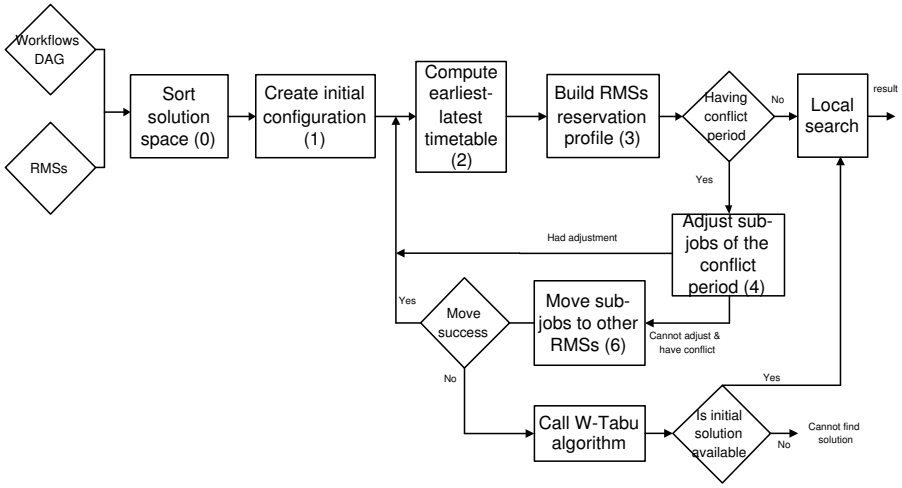


Fig. 4. Framework of the L-Map algorithm

Sj_ID	RMS-Rt	RMS-Rt	RMS-Rt
0	R1 - 16	R3 - 17	R2 - 16
1	R1 - 14	R2 -14	R3 - 14
2	R1 - 16	R2 - 16	R3 - 17
3	R1 - 16	R3 - 15	R2 - 17
4	R1 - 15	R2 - 17	R3 - 15
5	R1 - 14	R3 - 16	R2 - 16
6	R1 - 16	R3 - 16	R2 - 17

Table 2. RMSs candidate for each sub-job in cost order

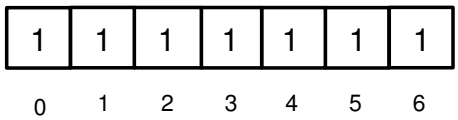


Fig. 5. The first selection configuration of the example

is described as a vector. The index of the vector represents the sub-job and the value of the element represents the RMS. The first configuration in our example is illustrated in Figure 5.

Step 2: As the runtime of each sub-job in the selected RMS was defined and the time to do data transfer equals zero, we can compute the earliest start time and the latest stop time of each sub-job using conventional graph algorithms. Following this procedure, we ensure that the Criteria 2 is met. In the case of our example, we assume that the user wants the workflow to be started at time slot 10 and stopped at time slot 85. The resulting Earliest-Latest timetable is shown in Table 3.

Step 3: For each RMS appearing in a configuration and each type of resource in the RMS, we build the resource reservation profile using the Earliest-Latest timetable. In this step, the runtime of the sub-job is computed from the earliest start time to the latest stop time. In our example, only RMS1 appears in the configuration. The CPU reservation profile of the RMS 1 is presented in Figure

Sj_ID	Earliest start	Latest stop
0	10	37
1	26	69
2	26	41
3	42	69
4	26	69
5	26	69
6	58	85

Table 3. The earliest-latest timetable

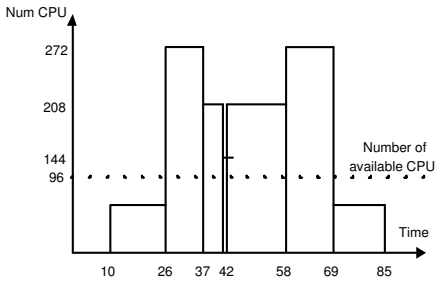


Fig. 6. Reservation profile of RMS 1

6. As can be seen from Figure 6, there are many conflict periods, in which, the number of required resources is greater than the available resources.

Step 4: We move sub-jobs out of the conflict period by adjusting the earliest start time or the latest stop time of the sub-jobs. One possible solution for our example is shown in Figure 7(a), where either the latest stop time of subjob1 is set to t1 or the earliest start time of subjob2 is set to t2. The second possible solution is to adjust both sub-jobs simultaneously as depicted in Figure 7(b). A necessary prerequisite here is that after adjusting, the latest stop time minus earliest start time of the sub-job is larger than its runtime.

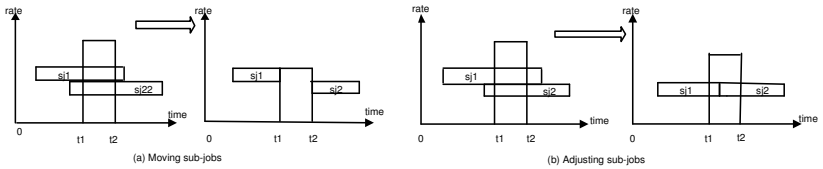


Fig. 7. Resolving the conflict period

Step 5: We adjust the earliest start time and latest stop time of the sub-jobs that are linked with the moved sub-jobs and then repeat step 3 and 4 until we cannot adjust any sub-jobs any more. The results of this step applied to our example, the CPU reservation profile of RMS 1, is depicted in Figure 8.

Step 6: If there are still some conflict periods after step 5, we have to move some sub-jobs contributing to the conflict to other RMSs. However, the resource, which has the conflict period, should be utilized as much as possible. Thus, the cost for using the resources will be kept at a minimum. This is a knapsack problem, which is known to be NP-hard. Therefore, we use the heuristic algorithm of Figure 9. This algorithm ensures that the remaining free resources are always less than the smallest sub-job. If a sub-job cannot be moved to another RMS, we can deduce that the Grid resource is busy and the w-Tabu algorithm [9] is invoked. w-Tabu algorithm maps a SLA-based workflow to Grid resources with

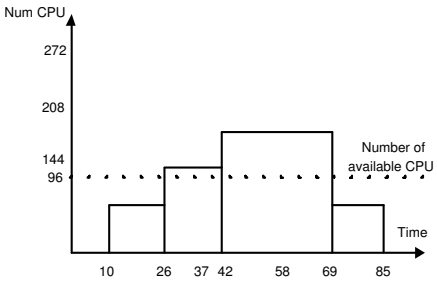


Fig. 8. Reservation profile of RMS 1 after adjusting

```
Select the most serious conflict period
Determine all sub-jobs contributing to the period
Sort those sub-jobs according to cost in descend order
For each sub-job in the list {
   If the resource free greater than the resource required by
the sub-job
      Let the sub-job stay in the RMS
      Update the number of resource free of the period
   Else
      Assign the sub-job to the next RMS in its sorted
candidate list          }
```

Fig. 9. Moving sub-jobs algorithm

makespan optimization. If the w-Tabu cannot find an solution, the algorithm will stop.

In the case of our example, the largest conflict period is 42-69 with allocations of sub-job 3, 5 and 4 (sorted in descending order according to the cost). Since we can fill the period with sub-job 3 only, sub-job 4 is moved to RMS 2 and sub-job 5 is moved to RMS 3. This step created a new configuration.

Step 7: As we created a new configuration, the process from step 3 to step 6 is repeated until there is no conflict period. This process will satisfy the conditions of Criteria 3, 4 and 5. After this phase, we have a feasible candidate solution.

Step 8: A local search procedure is used to improve the quality of the solution as far as possible. We search in the neighborhood of the candidate solution for a better solution. If a better solution has been found then it will play the role of the candidate. The process is repeated until we cannot find a better solution. More details about how this procedure meets all the criteria can be seen in [10].

4 Performance experiment

The goal of the experiment is to measure the feasibility of the solution, its cost, and the time needed for the computation. The hardware used for the experiments is rather standard and simple (Intel Duo 2,8Ghz, 1GB RAM, Linux FC5). To conduct the experiments, 18 workflows with different topologies, number of sub-jobs, sub-job specifications, and amount of data transferred between sub-jobs, are generated. These workflows are then mapped to 20 RMSs with different resource configurations and resource reservation profiles by 3 algorithms: L-Map, H-Map, and DBC. In the experiment, 30% number of RMS having CPU performance equals to the requirement, 60% number of RMS having CPU performance is 100% more powerful than requirement, 10% number of RMS having CPU performance is 200% more powerful than requirement. Along with the increasing in performance, the price for each CPU class is also increased. The runtime of each sub-job in each type of RMS is assigned by using formula 4.

$$rt_j = \frac{rt_i}{\frac{pk_i + (pk_j - pk_i) * k}{pk_i}} \quad (4)$$

with pk_i, pk_j being the performance of a CPU in RMS r_i, r_j respectively and rt_i being the estimated runtime of the sub-job with the resource configuration of RMS r_i. k is the speed up control factor. 60% number of sub-jobs having $k = 0.5$, 30% number of sub-jobs having $k = 0.25$, 10% number of sub-jobs having $k = 0$. The description about resource configurations and workload configurations can be seen at the address: http://it.i-u.de/schools/altmann/DangMinh/desc_expe1.txt. The final result of the experiment is presented in Table 4. Column Sjs (Sub-jobs) presents the number of sub-jobs within the workflows. The cost of the solution and the runtime for finding the solutions for each algorithm are recorded in column Runtime and column Cost.

	L-Map		H-Map		DBC	
Sjs	Cost	Runtime	Cost	Runtime	Cost	Runtime
Simple level experiment						
7	625.386634	1	625.386634	1	625.386634	0.5
8	749.706622	0.5	749.706622	0.5	755.306618	0.5
9	768.416626	0.5	768.416626	1	774.016622	0.5
10	830.883292	0.5	833.683290	1	833.683290	0.5
11	883.923294	0.5	883.923294	1	883.923294	0.5
12	936.393294	0.5	936.393294	2	936.393294	0.5
13	992.233294	0.5	992.233294	1	992.233294	0.5
Intermediate level experiment						
14	1044.093300	1	1044.093300	2	1049.693296	0.5
15	1169.953288	1	1169.953288	2	1175.553284	0.5
16	1294.273276	0.5	1294.273276	2	1294.333288	0.5
17	1353.649942	0.5	1353.649942	3	1353.649942	0.5
18	1477.969930	0.5	1477.969930	2	1477.969930	0.5
19	1501.396596	1	1501.396596	7	1504.196594	0.5
20	1560.773262	0.5	1560.773262	3	1560.773262	0.5
21	1584.153262	1	1584.153262	6	1584.153262	0.5
Advance level experiment						
25	1762.433256	2	1762.433256	5	1770.833250	0.5
28	1862.649928	3	1862.649928	7	1873.849920	0.5
32	2184.839906	4	2190.499914	10	2193.239900	0.5

Table 4. Performance experiment result

The experiments are divided into 3 levels. Within the simple level, 7 workflows with a number of 7 to 13 sub-jobs are mapped onto 20 RMSs. The result shows that the solutions created by different algorithms are identical for workflows with the same number of sub-jobs.

In the intermediate level experiment, we map 8 workflows that have 14 to 19 sub-jobs. The result of this experimentshow a difference in the quality of the solution found by the different algorithms. Method DBC, which do not use local search and need relative smaller runtime, found lower quality solution than other methods. Method H-Map found high quality solutions but needed more time than L-Map.

The advance level experiment mapped 3 workflows (with the number of sub-jobs in the range from 25 to 32). The result of this experiment shows that L-Map algorithm found out higher quality solutions than the DBC algorithm. It also found equal or even better solutions in a shorter time than the H-Map algorithm.

5 Conclusion

This paper has presented an algorithm, which performs a cost-efficient and fast allocation of workflows with light communication between sub-jobs onto Grid resources with respect to SLA-defined runtime constraints. In our work, the distinguishing factor is that the number of data to be transferred between sub-jobs is very small. Considering this factor allows to detect and resolve the conflict periods quickly. The simulation-based performance evaluation showed that the proposed algorithm creates solutions of equal or better quality than existing algorithms. Besides, it needed a significantly smaller computation time than the other two algorithms in our comparative study. These characteristics are positive factors for applying the method in real environments.

References

1. E. Deelman, J. Blythe, Y.Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh, S. Koranda: Mapping Abstract Complex Workflows onto Grid Environments. Journal of Grid Computing, Vol 1, no. 1, (2003) 25-39.
2. J. Cao, S. A. Jarvis, S. Saini, G. R. Nudd: GridFlow: Workflow Management for Grid Computing. Proc. 3rd IEEE/ACM Int. Symp. on Cluster Computing and the Grid, Tokyo, Japan, (2003) 198-205.
3. R. Lovas, G. Dzsa, P. Kacsuk, N. Podhorszki, D. Drtos: Workflow Support for Complex Grid Applications: Integrated and Portal Solutions, Proc. 2nd European Across Grids Conference, Nicosia, Cyprus, (2004).
4. A. Sahai, V. Machiraju, M. Sayal, L. J. Jin, F. Casati: Automated sla monitoring for web services. DSOM 2002, LNCS 2506, (2002) 28–41.
5. L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, H. Chang: QoS-Aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering, v.30 n.5, (2004) 311–327.
6. I. Brandic and S. Benkner and G. Engelbrecht and R. Schmidt: QoS Support for Time-Critical Grid Workflow Applications, Proc. e-Science (2005).
7. D.M. Quan, O. Kao: Mapping Grid job flows to Grid resources within SLA context. Proc. the European Grid Conference,(EGC 2005), LNCS 3470, (2005) 1107–1116.

8. D.M. Quan, O. Kao: On Architecture for an SLA-aware Job Flows in Grid Environments. Journal of Interconnection Networks, World scientific computing, (2005) 245–264.
9. D.M. Quan: Error recovery mechanism for Grid-based workflow within SLA context. To be published by International Journal of High Performance Computing and Networking (IJHPCN), (2007).
10. D.M. Quan, D.F. Hsu: Mapping Heavy Communication Grid-Based Workflows onto Grid Resources Within An SLA Context Using Metaheuristics. To be published by International Journal of High Performance Computing and Application (IJHPCA), (2007).
11. R. Buyya, M. Murshed, D. Abramson, and S. Venugopal: Scheduling Parameter Sweep Applications on Global Grids: A Deadline and Budget Constrained Cost-Time Optimisation Algorithm. Software: Practice and Experience (SPE), Vol 35, no 5, (2005) 491–512.
12. M. Hovestadt, O. Kao, A. Keller, A. Streit: Scheduling in HPC Resource Management Systems: Queuing vs. Planning. Proc. 9th Workshop on JSSPP at GGF8, LNCS 2862, (2003) 1-20.
13. D.P. Spooner, S.A. Jarvis, J. Cao, S. Saini, G.R. Nudd: Local grid scheduling techniques using performance prediction. IEEE Proc. Computers and Digital Techniques, (2003) 87–96.

Economics-Aware Capacity Planning for Commercial Grids

Marcel RISCH[1], Jörn ALTMANN[1,2], Yannis MAKRYPOULIAS[3], Sergios SOURSOS[3]
[1]*International University in Germany, School of Information Technology,*
Campus 3 Bruchsal, 76646, Germany,
Tel: +49 (0)7251 7000, Fax: + 49 (0)7251 700150, Email: marcel.risch@i-u.de
[2]*TEMEP, School of Engineering, San 56-1, Sillim-Dong,*
Gwanak-Gu, Seoul, 151-742, South-Korea
Tel: +1 510 972 3062, Fax: +1501 641 5384, Email: jorn.altmann@acm.org
[3]*Athens University of Economics and Business, Dept. of Informatics,*
76, Patission str, Athens, 10434, Greece
Tel: +30 210 8203693, Fax: + 30 210 8203693, Email: {makrupoul06, sns}@aueb.gr.

Abstract: Currently, capacity planning is fairly simple, due to the few options that are available. With the advent of Grid markets, this discipline for analyzing resource purchases must be adapted. A commercial Grid provides many different resource types at variable prices, making capacity planning more complicated than it currently is. In this paper, we describe the functionality of an online Grid Capacity Planning Service, which helps companies with little IT expertise to make use of the Grid in a cost-effective manner. The requirements of the capacity planning service are derived in part from a survey carried out among SMEs in the region of the German city of Bruchsal. Using the requirements, we identified all the necessary information from the Grid and we designed and implemented parts of the capacity planning service.

1. Introduction

The main foci of Grid research fall into two categories: In the first category are the research and development projects, which aim at developing Grid architectures. These include Globus [1], GRIA [2], Gridbus [3], glite [4] and NextGRID [5]. In the second category are the business-related projects, which analyze and develop business models for Grid systems. These include, amongst others, AssessGrid [6], Biz2Grid [7] and BeInGrid [8].

However, the commercial Grids, which are slowly developing, do not seem to rely on the principles developed by these projects. Instead, existing commercial Grids have been developed by companies, which have created their own approaches to Grid computing, such as the Amazon.com EC2 [9], the Sun Grid [10], and the Tsunamic Technologies Grid [11]. Some of these services have started to become very popular and, thus, prove that the idea of commercial Grid computing is indeed acceptable to resource providers and buyers [12]. This is illustrated in SecondLife Blog [13], which illustrated that for short demand peaks, the Amazon service was cheaper than installing additional in-house bandwidth capacity.

Existing research in commercial Grids has largely failed to consider the economic aspects of Grid computing. It has been tacitly assumed that, by providing the proper technical environment, customers would be convinced of the advantages of Grid technologies. This shortcoming of existing Grid research has been identified by the GridEcon project [14], which started out with the premise that economic considerations complement technical solutions. A number of economic-enhanced support services have been identified which are needed for a commercial Grid to function. These services include

various brokers but also a capacity planning service, which helps customers to determine their resource needs.

This paper shows the steps towards the development of a Grid Capacity Planning Service, which will help Grid users to plan their capacity requirements. First, we will define the term 'capacity planning', since capacity planning has lost some of its importance in data center planning and since there are many different definitions for this task. Second, we demonstrate the need for a capacity planning service in a commercial Grid environment through the analysis of the differences between traditional capacity planning and Grid capacity planning as well as through the analysis of a qualitative survey. The survey addressed SMEs from diverse fields such as banking, software development, and car sales. From the survey results, we determined the requirements that SMEs have towards Grid markets and towards capacity planning in particular. Based on these results, the requirements of the capacity planning service are determined and an architecture of a Grid Capacity Planning Service is designed and developed.

The structure of the paper is devised as follows: In Section 2, capacity planning is put into context with other data center tasks. In Section 3, the differences between traditional and Grid capacity planning are explained and, from these differences, the requirements for a Grid capacity planning service as well as the design of the service are derived. The Short-Term Capacity Planning Service is then explained in more detail in Section 4, before the paper is concluded in Section 5.

2. Capacity Planning Definition

2.1 Capacity Planning Definition

In [15], it has been remarked that the term "capacity planning" is frequently used but rarely defined. To correct this deficiency, we will start by defining the term, using the definition given by IBM in [16]:

> "*Capacity Planning encompasses the process of planning for adequate IT resources required to fulfill current and future resource requirements so that the customer's workload requirements are met and the service provider's costs are recovered.*"

This definition allows us to categorize the users of capacity planning into two groups: customers and providers. While some research has been done on the provider's capacity planning problems [17][18], no research has been done, as of yet, on the customer's need for capacity planning in a Grid environment.

According to the definition, the following three tasks are at the heart of the capacity planning process: (1) Monitoring the current resource utilization rate and the application response times; (2) Estimating the future resource requirements of applications; (3) Cost monitoring to ensure that a company does not overspend. Based on these tasks, four courses of action are open to companies: (1) purchasing in-house resources, (2) renting or leasing in-house resources, (3) doing nothing, or (4), in the case of the existence of a commercial Grid, purchasing Grid resources.

2.2 Conceptual Classification of Capacity Planning

The capacity planning process should guarantee that a basic mapping of applications to resources (i.e. resource allocation) is always possible. This implies that resource allocation is a sub-task of the capacity planning process. At the same time, the capacity planning process should ensure that resources have similar load levels, which implies that load balancing is also a subtask of the capacity planning process. Load balancing and resource allocation already have one task in common, namely monitoring.

Economic aspects have always been a part of capacity planning. Since data centers have budget constraints, the IT personnel needs to determine which resource has the best value

for the price. These issues are also faced in a commercial Grid, where the IT personnel has to compare different Grid alternatives. As we will outline in the next section, the task of choosing the optimal resource is even more complex because of the usage-based pricing structure of commercial Grids. It forces the data center personnel to predict the resource usage very precisely, in order to achieve low costs for Grid usage.

3. Towards a Grid Capacity Planning Service

3.1 Complexity of Decision Making in Capacity Planning

This section demonstrates why capacity planning is very important in a Grid environment. It also discusses the differences between traditional capacity planning and Grid capacity planning. For this analysis, we assume that a functioning Grid economy exists, that is, prices are determined according to the supply and demand situation. The currently existing Grid computing offers (such as Amazon EC2, Sun and Tsunamic) have fixed prices.

3.1.1 Effort of Capacity Planning

Capacity planning is not popular with companies, since the effort does not bear any relation to the expected benefits, as can be demonstrated with the following example:

We assume that an employee incurs costs of $4500 per month (salary and human resource management) to a company. Thus, one week of work on planning the computing resources costs the company about $1000. This cost must be added to the cost of the computing resources. If this company is a small company, the actual savings through an accurate capacity planning procedure is very low or even non-existent. Consequently, capacity planning is not widely used. This fact could be verified during a qualitative survey of SMEs in the Bruchsal region (Germany): When asked to describe their capacity planning process, companies stated that they used their “gut feeling” to decide which resource to purchase. Other companies simply bought one of the most powerful computing resources available to ensure that it would be able to run future applications, which are expected to require more powerful resources.

In a commercial Grid environment, provider companies have even more capacity planning inputs to consider to determine the best resource allocation. Thus, the capacity planning process becomes even more expensive. This process is further complicated by the fact that Grid users may be willing to sell excess resources on the Grid. In this case, the expected income must be taken into account when calculating the Grid usage costs. This increased complexity will require the data center staff to spend more time on the capacity planning process, which in turn reduces the benefit of capacity planning further.

3.1.2 Resource Diversity

In traditional capacity planning, the IT personnel only has to select new hardware from the resources currently available in the market. Although not standardized, these resources usually have similar features and thus pose few difficulties for professional staff. Therefore, should a resource be added or replaced, the data center staff will be able to find a similar resource that can perform the job adequately well, without having to perform any performance testing. This has also been reported in the survey: Companies stated that new resources are better in every aspect than existing resources, so that all existing applications and future applications (with increased resource requirements) can still be executed.

In a Grid environment, however, the diversity of resources is much greater, since providers may offer any kind of resource (e.g. virtual machines of old computers).

3.1.3 Price Volatility

The current computing resource market is fairly static in that resource prices do not change frequently. Since current resources are usually bought for in-house installation, price variations only occur because of special offers or economies of scale. Therefore, the capacity planning team does not need to rush the capacity planning process to avoid changing prices. Even if the capacity planning team decides to make use of current commercial Grid resources (e.g. Amazon [9], Sun [10], Tsunamic [11]), those prices, although high, remain unchanged [19].

With the advent of commercial Grids, which sell resources at competitive prices, prices will change according to the variation in supply and demand. The capacity planning team has to consider these price fluctuations and has to predict how the prices will develop. Furthermore, with changing prices, the timing of purchases may become a relevant parameter in the capacity planning process: The demand peaks have to be analyzed with respect to the market prices, in order to determine whether the demand peaks coincide with times of high Grid prices.

3.2 Requirements of a Grid Capacity Planning Service

In order to be accepted by Grid users, a Grid Capacity Planning Service (GCPS) should fulfill a number of requirements and provide a set of basic functionality.

The functionality that should be offered by the GCPS is the monitoring service. It monitors Grid and in-house resources. If a performance requirement is no longer met (e.g. an exceeded response time limit or exceeded load level), the GCPS triggers the resource analysis process in order to determine a course of action to satisfy all user demands.

During the resource analysis process, the Grid Capacity Planning Service should take the user's in-house resources into account and should ensure that these are used. Since these resources have been purchased and have been installed, they should be used as much as possible to reduce costs. However, if the user is willing to sell certain resources on the Grid, the GCPS has to take this into account as well. There may be cases in which it is advantageous to sell some of the in-house resources on the Grid.

When the GCPS has determined which courses of action are viable, it should give a ranked list of options to the user, who can then decide which of these options should be executed. This final selection step allows the user to remain in control of his budget and to ensure that tacit requirements are also met.

Furthermore, the GCPS should perform application-resource-mappings. Since the GCPS needs to determine the requirements of applications under different circumstances, the GCPS has to run performance tests for applications on various resource types. At the same time, this procedure will have to be performed in a cost-effective manner. In a commercial Grid environment in which prices fluctuate, the GCPS should perform these actions during times of low resource prices. The results of the performance tests and the thereby derived requirements for each application should be stored locally within the GCPS to ensure that all information is readily available.

Finally, the Grid Capacity Planning Service should perform as many of these actions as possible automatically, i.e. without the help of a human operator. There are two arguments that count against human involvement: the first is that people are slow and error prone and thus the result generated by the GCPS would not be as accurate. The second reason is that experienced and specialized employees are expensive and, thus, would cause higher costs.

3.3 A Grid Capacity Planning Service Model

To fulfill all requirements, we envision a Grid Capacity Planning Service (GCPS), which consists of two distinct parts that work in concert:

- Part 1: Long-Term Capacity Planning Service (LTCPS): This is an online service, which performs the long-term analysis of the user's resource situation. It takes input parameters (e.g. information about in-house resources, application information, and user requirements, and information from the Short-Term Capacity Planning Service) before determining a ranked list of possible courses of action for in-house resource purchases and rentals, and Grid resource purchases and sales.
- Part 2: Short-Term Capacity Planning Service (STCPS): This service suggests to the user the number of machines required to meet his short-term performance or economic goals, over a specific time period. To do so, it uses information on past usage of Grid resources, considers characteristics of the application the user wants to run on the Grid and estimates the application's load that is expected for the requested time period.

These two capacity planning services complement each other, since one addresses the short-term problems while the other addresses the medium- to long-term problems. The two services will be described in more detail in the following two sections.

3.3.1 The Long-Term Capacity Planning Service

Since the LTCPS should plan as far into the future as possible, it requires a lot of information, which can be grouped into three categories:

The first category includes information about user-owned resources, user-owned applications, as well as user requirements. Information about user-owned resources and applications describe which resources are installed in-house and which applications are to be run. The information about user requirements is more complex. For example, many companies have sensitive data, which should not be processed by applications beyond company boundaries. These applications are not permitted to run on the Grid, since this would violate company policy (i.e. a user requirement). Another example for a user requirement is the maximum expenditures for Grid resources, runtime limits, the response time limits, or the fact that the in-house resources must be used to their full extend.

The second category consists of application requirements. These can be divided into minimum and optimal requirements and include items such as hardware requirements and software requirements. The software requirements provide information about additional software that must be available on a machine. The software requirements also include requirements for communication with other software components (e.g. communication with other sub-jobs within a workflow). Furthermore, the application requirements should also include the usage frequency of the application and the average usage duration.

The third category contains information about the Grid market and is collected by the LTCPS. It comprises the current and past prices of Grid resources, the availability of suitable Grid resources, as well as the prices for in-house resources. This last point is still a challenge, since many resource vendors do not use any standard for composing computers that would allow a potential buyer to determine computer prices automatically.

Next, these three sets of information are used to determine a mapping of applications to resources in such a way that all user requirements are met. The result can fall into one of the following categories: in-house resources are sufficient, purchase Grid resources, purchase in-house resources, sell in-house resources on the Grid, or purchase both in-house and Grid resources. The operation of the LTCPS can be seen in Figure 1 below.

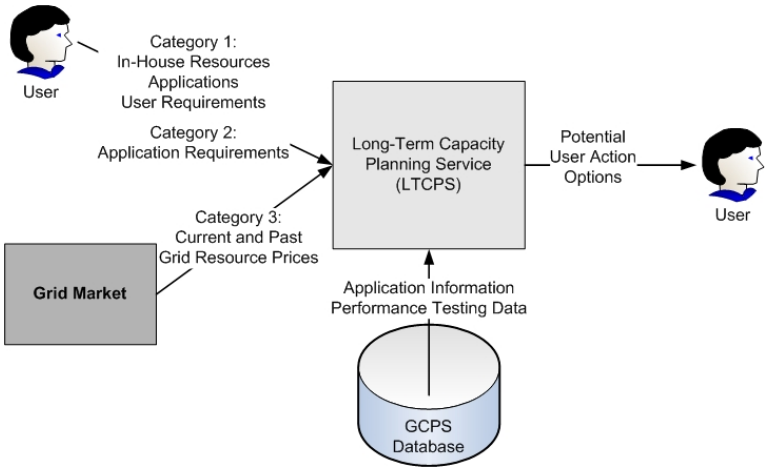


Figure 1: The Long-Term Capacity Planning Service

3.3.2 The Short-Term Capacity Planning Service

The STCPS takes input from the user with respect to the performance and cost-minimization objectives. Then, it proposes the number of machines to be purchased on the Grid, in order to fulfill the user requirements.

We assume that performance objectives are expressed in terms of time delay (i.e. the response time of an application for serving a request). Users (e.g. application providers) find it easy to express their requirements in terms of delays, since the system response time can be experienced by users.

For cost-minimization purposes, further information is required. First, the information about the cost of purchasing a single machine on the Grid market is necessary. Second, it is required that the user provides a cost function, i.e. a function that relates the experienced delay per application request with a monetary value, representing the cost that is incurred to the provider per millisecond of delay per request.

In addition to the information about the objectives of the user, the STCPS needs to have access to monitoring data about the resource load. The monitoring data is needed to make the necessary estimations and predictions of the load (number and type of requests) that the application will face in the future time period for which the plan is required. All these input are summarized in Figure 2 below.

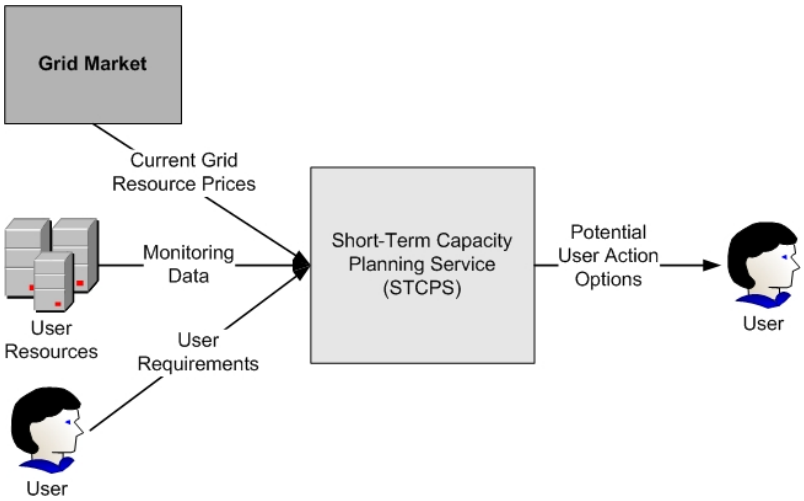


Figure 2: The Short-Term Capacity Planning Service

4. A Closer Look at the Short-Term Capacity Planning Service

Within the GridEcon project [14], we have designed and implemented the Short-Term Capacity Planning Service. In the following sections, we will present the architecture of this system and describe the functionality of the components.

For the STCPS to function, we have to collect monitoring data about every application that runs on the Grid and to be able to categorize applications according to some common characteristics. We assume that every Grid system provides some public interfaces for accessing monitoring data. Using these interfaces, we can obtain all the needed information and store it to a local database, namely the History Component.

After collecting the necessary data, we run a classification algorithm, executed by the Clustering Component. Clustering is necessary for the STCPS, since it allows working with an aggregated set of application-related data. The criterion according to which the application is classified is the execution time of a single request to a virtual machine on the Grid. Our assumption is that applications with similar execution times per request are similar enough to be considered as identical applications. For sake of simplicity, we use clustering with only two classes as an output. This parameter of the system can be changed to allow clustering results with a higher number of classes.

After having aggregated the historical data about the load, we can estimate the load expected for the time period defined by the user. This task is assigned to another module, the Workload Predictor Component. It takes into account the load of the previous hours, along with the load encountered at the same day and time over the past few weeks. Thus, we capture the current trend of the load as well as the behavior that appears periodically.

After the workload prediction is complete, the Decision Support Component has all information necessary to propose to the user the minimum number of virtual machines needed to meet the requirements. To give the user a better recommendation, we have implemented two models: an M/M/k queuing model and a Support Vector Machine model. Both models give answers to the same questions, e.g. how many machines are required such that the application A provides an average service time of Y milliseconds for the next 2 hours? Or, how many machines are required to minimize the provisioning cost, provided that each virtual machine costs x Euros per hour and the cost function is of a certain shape?

All the aforementioned components, along with their internal and external interactions, are depicted in Figure 3.

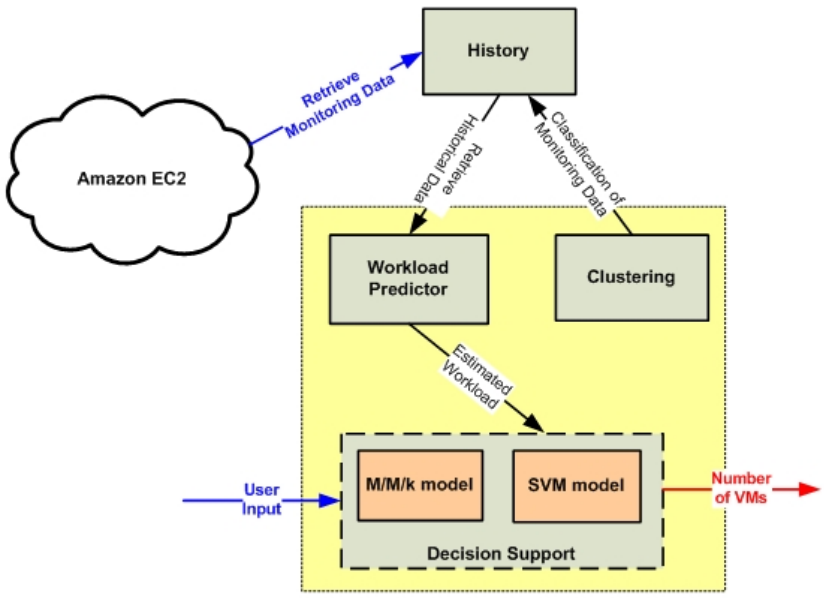


Figure 3: The Architecture of the STCPS

The system is already implemented using the .NET platform and relying on the Amazon EC2 environment. For our tests, we also designed and implemented an application-level monitoring module. The test application that is executed on EC2 is a Web Server with different JSP web pages that emulate the different Web applications. We are currently testing the application and the various approaches used in the design, in order to evaluate the contribution of such a system.

5. Conclusions

In this paper we have defined the term "capacity planning" and have placed it in a conceptual context for Grid computing. Furthermore, we have shown that capacity planning is rarely used in companies today and that it is more complex in a Grid environment. Due to the complexity, we believe that a Grid Capacity Planning Service (GCPS) is necessary for not only a successful Grid usage but also for making Grid computing widely used.

The GCPS described in this paper has two parts: the Short-Term Capacity Planning Service and the Long-Term Capacity Planning Service. The first is responsible for ensuring that all applications are running as required and will give advice regarding additional resources if they become necessary for a short time period. The latter is responsible for long-term planning of data centers and takes into account the resource requirements of all applications, the available in-house resources, and the user requirements.

Since only an outline of the LTCPS and a basic testing implementation of the STCPS have been developed, several issues still need to be addressed in implementing the GCPS: Firstly, the efficiency of this service needs to be investigated. Since performance testing is expensive, the GCPS needs to perform as little performance testing as possible without ignoring available resource types. Secondly, the GCPS needs to take into account that an application can have different resource requirements depending on how it is used. Finally, the GCPS has to ensure that it performs all capacity planning tasks quickly, even if the user has many courses of action open. This may require storing common resource allocation solutions but, at the same time, avoiding the storing of excessive amounts of data.

References

[1] The globus alliance, http://www.globus.org/, 2008.
[2] Gria, http://www.gria.org/, 2008.
[3] Gridbus, http://www.gridbus.org/, 2008.
[4] gLite, http://glite.web.cern.ch/glite/, 2008.
[5] NextGRID: Architecture for Next Generation Grids, http://www.nextgrid.org/, 2008.
[6] AssessGrid, http://www.assessgrid.eu/, 2008.
[7] Biz2Grid, http://www.d-grid.de/index.php?id=407&L=1, 2008.
[8] BeInGrid, http://www.beingrid.eu/, 2008.
[9] Amazon Elastic Compute Cloud (Amazon EC2), http://www.amazon.com/gp/browse.html?node=201590011, 2008.
[10] Sun Grid, http://www.sun.com/service/sungrid/index.jsp, 2008.
[11] Tsunamic Technologies Inc., http://www.clusterondemand.com/, 2008.
[12] Techcrunch: http://www.techcrunch.com/2008/04/21/who-are-the-biggest-users-of-amazon-web-services-its-not-startups/, 2008.
[13] SecondLife Blog, http://blog.secondlife.com/2006/10/26/amazon-s3-for-the-win/, 2008.
[14] GridEcon, http://www.gridecon.eu, 2008.
[15] Cortada, J.W.: Managing DP Hardware. Capacity Planning, Cost Justification, Availability and Energy Management. Prentice-Hall, Inc., Englewood Cliffs (1983).
[16] IBM: A Statistical Approach to Capacity Planning for On-Demand Computing Services, http://domino.watson.ibm.com/comm/research.nsf/pages/r.statistics.innovation2.html, 2008.
[17] Siddiqui, M., Villazon, A., Fahringer, T.: Grid Capacity Planning with Negotiation-based Advance Reservation for Optimized QoS. In: Supercomputing, 2006. SC '06. Proceedings of the ACM/IEEE SC 2006 Conference, IEEE, 2006.

[18] Borowsky, E., Golding, R., Jacobson, P., Merchant, A., Schreier, L., Spasojevic, M., and Wilkes, J.: Capacity planning with phased workloads. In: Proceedings of the 1st international Workshop on Software and Performance. WOSP '98, pp. 199-207 ACM, New York, NY, 1998.
[19] Risch, M., Altmann, J.: Cost Analysis of Current Grids and its Implications for Future Grid Markets. In: Proceedings of the Grid Economics and Business Model Workshop. Gecon 2008, pp.13-27. Springer LNCS. Heidelberg, 2008.