Strategic Negotiation Models for Grid Scheduling

von der Fakultät für Elektrotechnik und Informationstechnik

der

Universität Dortmund

Gehnemigte

DISSERTATION

zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften

von

M.Sc. Jiadao Li

Tag der mündlichen Prüfung: 18. Oktober 2007
Hauptreferent: Prof. Dr.-Ing. Uwe Schwiegelshohn
Korreferent: Prof. Dr. habil. Sergei Gorlatch

I would like to dedicate this thesis to my parents.

Acknowledgements

I'd like to thank all of the people who have supported and help me to make this thesis possible. Firstly and foremost, I am deeply indebted to my supervisors, Prof. Dr.-Ing. Uwe Schwiegelshohn and Dr.-Ing. Ramin Yahyapour. During the last 3 and a half years, they have always encouraged and motivated me, given valuable advices to me. I owe my whole research progress to their efforts. I also want to thank Prof. Dr. habil. Sergei Gorlatch for his helpful support.

Further, I would like to thank for the help of the all of the other colleagues in the information technology sector, institute for robotics research. They provided me a very pleasant working atmosphere. It is a cherishable memory to work with them in my life. Also I would like to thank for the finical support from the Graduate school of production engineering and logistics, university of Dortmund. The help from the colleges from graduate school is highly indebted.

Finally, I would like to thank my parents for their love and their support at all times.

Abstract

One of the key requirements for Grid infrastructures is the ability to share resources with nontrivial qualities of service. However, resource management in a decentralized infrastructure is a complex task as it has to cope with different policies and objectives of the different resource providers and the resource users. This problem is further complicated due to the diversity of the resource types and the heterogeneity of their local resource management systems. Agreement-based resource management can be used to address these issues because in the negotiation process of creating such bilateral service level agreements (SLAs) between Grid parties, the different polices of the resource providers and the users will be abstracted and observed. Such negotiation processes should be automated with no or minimal human interaction, considering the potential scale of Grid systems and the amount of necessary transactions. Therefore, strategic negotiation models play important roles. In this thesis, we have made several novel research contributions which are as follows:

- An agreement based resource management approach is analyzed. Requirements for the automatic negotiation problems in Grid computing are introduced. Furthermore, related work in the areas of economics and agent communities are investigated.
- Several negotiation models and negotiation strategies are proposed and examined. Simulation results demonstrate that these proposed negotiation models are suitable and effective for Grid environments.
- Firstly, a strategic negotiation model using time-based negotiation strategies is proposed and evaluated using discrete event based simulation techniques.

- Secondly, time-based negotiation strategies are quite limited in the dynamically changing Grid environment because they are quite simple and static; so learning based negotiation strategies are investigated and evaluated, which are quite flexible and effective in the dynamically changing Grid environment. Also we adopted negotiation strategies considering opportunistic functions for Grid scheduling.
- Thirdly, it is usually necessary that resources from different resource providers are co-allocated to satisfy the complex requirements of the users, so a strategic negotiation model supporting co-allocation and the tradeoff between "first" and "best" agreements in the Grid computing is also proposed and evaluated.
- Finally, the contributions of the current research work to the WS-Negotiation protocol are analyzed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Resource Management in Grid Computing

Grid computing [1, 2] is considered a cornerstone of next generation distributed computing, which is defined as "coordinated resource sharing and problem solving in dynamic, multi-institutional collaborations". Current Grid computing infrastructure is built in accordance with the service oriented architecture (SOA [2, 3]) paradigm. A service-oriented architecture is one in which all entities are services. A service is an entity that provides some capability to its clients by exchanging messages and it is defined by identifying sequences of specific message exchanges that cause the service to perform some operations. By thus defining these operations only in terms of message exchange, we achieve great flexibility in how services are implemented and where they are located.

In this service oriented architecture, the services [4] may include both traditional resources (e.g., computational services offered by a computer, network bandwidth, or space on a storage system) and virtualized services (e.g., database, data transfer, simulation), which may differ in the functions they provide to users but are consistent in the manner in which they can deliver those functions across the network. These services are loosely coupled in the SOA.

Resource management is an important issue in Grid environment which refers to the operations used to control how resources are made available by providers to consumers. Resource management is commonly used to describe all aspects of the process of locating various types of capability, arranging for their use, utilizing them, and monitoring their states. Grid scheduling involves three main phases [5]: **resource discovery**, which generates a list of potential resources; **information gathering** about those resources and selection of a best set; and **job execution**, which includes file staging and cleanup. There are several actions involved in each phase which are shown in Picture 1.1.

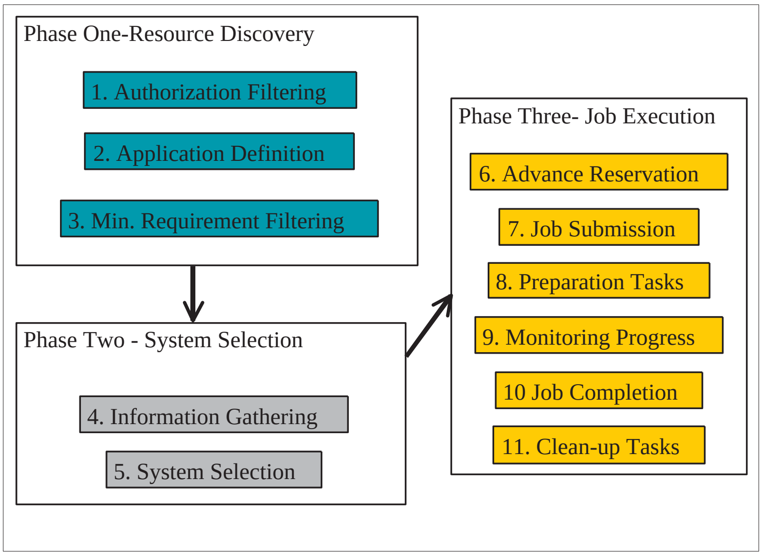

Figure 1.1: Actions Involved In Grid Resource Management

In the Grid scenario, the typical characteristics of *resource providers* are as follows:

1. There is no central control available among them which means a higher-level resource management system can not have full control over the resources of different providers and can not obtain the complete information of resources.

2. The resource providers have individual and quite different objectives and policies which have to be considered. For instance, some resources can only be used by certain users or during specific times.

3. Their local resource management systems are heterogenous. Some examples of such local resource management systems are: Platform's Load Sharing Facility [6], Portable Batch System [7], IBM's LoadLeveler [8], Sun's Grid Engine [9], etc.

4. Their resources are quite heterogenous. Their hardware and operating systems can be from different vendors.

The typical characteristics of *resource consumers* are:

1. The request of a consumer is usually non-trivial. It is often necessary that different resource providers cooperate and coordinate together to satisfy the complex request of a consumer (Co-allocation or workflow support is needed).

2. The consumer wants to obtain some guarantees for the quality of service from service providers. That is, for instance, the service properties (cost, waiting time, etc.) are known in advance.

3. Similar to the resource providers, the resource consumers also have individual and quite different requirements and objectives. Some consumers want to get cheaper services, while others may prefer earlier execution.

The resource management problem in this decentralized scenario is different from conventional resource management problems. The conventional centralized scheduling approach is not well suited to solve this problem, as it usually assumes a system model, which is an abstraction of the underlying resources in a single administrative domain. The resource management system can obtain the complete and up-to-date information of resources and can autonomously *schedule* resources. The resource management system is constructed to optimize some system-wide specific objectives [10, 11, 12].

However, in the Grid environment, the resource management spans different administrative domains in a decentralized manner. So it is difficult for the resource management system to obtain a global system model and get complete and up-to-date information of resources. In general, it is not feasible to *schedule* all these resources owned by different resource providers using conventional centralized approaches. Typically the resource management problem in Grid is a hierarchical [5] system which is comprised of higher-level resource management systems and lower-level resource management systems. The higher-level resource management system does not have its own resources and can not control over the resources of different resource providers and it acts as the intermediary between the end consumers and lower-level resource management systems. The lower-level resource management system is the conventional resource management system which is assumed to have full control over its local resources.

As introduced before, a resource user typically expects a certain service quality to be provided by the resource owners. As in Grid computing usually autonomous parties interact, the a priori information of quality of service (QoS) becomes a crucial requirement for resource management in this scenario. To this end, prior to service execution the parties have to negotiate towards service level agreements (SLAs) that define what kind of service will be provided and what the obligations of the user will be.

Of course, the resource providers and the resource consumers have typically conflicting objectives which need to be considered during the negotiation. The whole task of negotiation is challenging as the resources are heterogeneous and the service provisioning is not a standardized good but depends on the individual requirements and preferences of the user for a particular task. During the negotiation process, the conflicts of the different objectives and policies between the resource users and resource providers must be reconciled. However, this process should be automated as it cannot be expected that service providers and consumers have the ability to pursuit the negotiation by themselves. For efficient resource management in distributed systems, such as Grid computing, this task will be frequently performed and highly relies on the dynamic resource conditions. Here, **suitable strategic negotiation models** are required that take the different policies and objectives into account and which produce suitable service level agreements in reasonable time with minimized or even no user and provider interference.

This thesis aims to develop the negotiation models supporting the automatic negotiation for Grid scheduling and investigate the negotiation strategies in the proposed negotiation models.

1.2 Contributions

The main contributions of the thesis work are as follows:

- The agreement based resource management approach is analyzed. Requirements for the automatic negotiation in Grid computing are introduced. Related work in the areas of resource management in Grid computing, strategic negotiation research in economics and agent communities are investigated.
- Several negotiation models and negotiation strategies are proposed and examined. Simulation results demonstrate that these proposed negotiation models are suitable and effective for Grid environments.

  - Firstly, a strategic negotiation model using time-based negotiation strategies is proposed and evaluated using the discrete event based simulation techniques.
  - The time-based negotiation strategies are quite simple and static. Therefore, they are quite limited in the dynamically changing Grid environment; so learning based negotiation strategies are investigated and evaluated, which are quite flexible and effective in the dynamically changing Grid environment. Also we adopted and evaluated the negotiation strategies considering opportunistic functions for Grid scheduling
  - Furthermore, it is usually necessary that different resource providers co-allocate together to satisfy the complex requirements of the users, so a strategic negotiation model supporting co-allocation in the Grid computing is also proposed and evaluated in this thesis.

- Finally, the contribution of the current research work to the WS-Negotiation protocol is analyzed.

1.3 Structure

The thesis is structured in 8 chapters. In Chapter 2, the requirements for the automatic negotiation for Grid computing are introduced. In Chapter 3, the strategic negotiation model which supports the simple scheduling scenario are proposed and evaluated. In Chapter 4, learning based negotiation strategies are proposed and evaluated. In Chapter 5, the negotiation model which supports the co-allocation problem is presented. In Chapter 6, the negotiation strategics considering the opportunity functions are evaluated. In Chapter 7, the contributions of current work to the WS-Negotiation protocol are analyzed. Chapter 8 summarizes the whole thesis and gives possible future directions.

Chapter 2

Automatic Negotiation for Grid Scheduling

As introduced in Chapter 1, the Grid resource management in the decentralized Grid environments is challenging. The conventional centralized approaches can not be easily used to schedule the resources belonging to different administrative domains. Therefore, agreement based resource management [4] is typically considered as a suitable approach for this scenario. In this approach, various resource management activities can be represented using the terms of agreements. In this chapter, the agreement based resource management approach is analyzed. Requirements for building the agreement based resource management in Grid computing are introduced. Related works in the areas of resource management in Grid computing, automatic negotiation techniques in economics and agent theory communities are introduced. Strategic negotiation models which are quite important for supporting the automatic negotiation are explained.

2.1 Agreement Based Resource Management

In order to influence the resource usage, a resource consumer needs to understand and control resource behavior, often requiring assurances or guarantees concerning the level and type of service being provided by the resource. Conversely, the resource provider wants to maintain local control over how the resource can be used and how much service information is exposed to the consumer of the resource [4]. A common means for reconciling these two competing demands is to negotiate an agreement (sometimes called a Service Level Agreement, or SLA), by which a resource provider "contracts" with a client to provide some measurable capability. Agreements explicitly state the terms between a resource user and resource provider allowing clients to understand what to expect from resources and also the penalties if either of them violates the negotiation terms. An agreement can be viewed as more than a simple statement of terms of performance. Rather, an agreement can be viewed as a statement of common policy terms to be honored by the provider and consumer of the agreement. As such, an agreement provides a powerful mechanism for virtualizing or abstracting a resource. Furthermore, the agreement can also be multilayered based, as higher-level services such as a super-scheduler (Grid level scheduler) might act as intermediaries between an end-user and a local scheduler adding an additional layer as well as serving to broaden the scope of users' requests. Services might thus be composed of different levels and the acquisition of a service by some end-user requires the transitive access to all agreements.

There can be different kinds of requests or application scenarios from the users' side in Grid computing, some of them can be simple job execution, while others can be complex workflows or need the co-allocation between different resource providers. These job types can be easily supported by the agreement based resource management approach.

Negotiation is the de-facto means of creating the service level agreement between the resource consumers and the resource providers. The whole task of negotiation is challenging as the resources are heterogeneous and the service provisioning is not a standardized good but depends on the individual requirements and preferences of the user for a particular task. During the negotiation process, the conflicts of the different objectives and policies between the resource users and resource providers must be reconciled. The negotiation process in a Grid computing environment should be done automatically and transparently with the growing scale in Grids [13]. For efficient Grid computing, this task must be performed very frequently and it is highly affected by the dynamic resource conditions. Thus, during the negotiation process, every user or resource provider will have an agent or resource broker as a negotiation wrapper which will act on behalf of the participant.

In order to automate the negotiation process, suitable negotiation models are required that take the different policies and objectives of the resource providers and resource users into account and produce suitable service level agreements in reasonable time with minimized or even no user and provider interference. Before the negotiation process, the negotiation models between the user and the provider should be provided.

Currently, there is no mature and accepted negotiation model available for the Grid computing scenario.

2.2 Requirements for Agreement Based Resource Management in Grid Computing

There are many Grid projects worldwide in which several Grid infrastructures or Grid middleware tools are implemented, e.g., Globus [14], glite [15]. These Grid middlewares have identified and provided most of the functions to build the Grid systems. In this section, the requirements for implementing the agreement based resource management system are explained.

2.2.1 Authorization and Authentication

Authentication and authorization are essential for building the Grid infrastructure. To enable the scheduler or job agent to act on behalf of the user the respective rights have to be delegated from the user to the agent. Authentication mechanisms are required so that the identity of individuals and services can be established. Service providers must implement authorization mechanisms to enforce policy over how each service can be used.

2.2.2 Job Requirement and Resource Description

The requirements of the jobs are quite different. Users submit their requirements for the resources by different attributes, for example, by capability, quality, or configuration. These requirements must be expressed through a resource description language. Furthermore, in order to make the resources' configurations or capacities between different domains and the different, complex job requirements to be understood by each other, interoperable and standard job requirement and resource description languages should be defined. These languages [4] enable the resource consumer to describe what capabilities are desired, and what they will be used for (e.g., job configuration). They also make it possible for a resource provider to describe the capabilities that it can offer, and under what terms the resource will be offered. Therefore, a standard and dynamically extensible job/resource description language should be defined. These languages should be extensible in different domains and scenarios. Some examples of such languages are: the Globus resource specification language (RSL) and condor classified Ads (ClassAds) [16], job submission description language (JSDL) [17].

2.2.3 Information Service

Information service is an important component in Grid infrastructure which provides the needed information for the resource providers, schedulers and the users, etc. There can be relatively static information (e.g., the total CPU number of the resources, the hardware and software configuration) which can be procured through the resource discovery and dynamic information (e.g., the current utilization rate of some resource, the availability of special services) which usually can be obtained in the process of scheduling or negotiation. An example information service is the Monitoring and Discovery System (MDS), which is a suite of web services to monitor and discover resources and services on Grids. This system allows users to discover what resources are considered part of a Virtual Organization (VO) and to monitor those resources. MDS services provide query and subscription interfaces to arbitrarily detailed resource data and a trigger interface that can be configured to take action when pre-configured trouble conditions are met [18]. Some of the information services use the relational database model and the SQL query language (e.g., R-GMA [19]).

2.2.4 Resource Discovery and Selection

The first stage in any scheduling interaction involves determining which resources are available to a given user. Resource discovery is the process of querying the distributed state of the Grid to identify those resources whose characteristics and states match those desired by the resource consumer. Usually after the resource discovery and resource pre-selection according to static resource information (e.g., the maximum provided CPU nodes) of the resources and the requirements of the job users, several candidate resources are selected for the job users.

2.2.5 Agreement Negotiation

As introduced before, negotiation is the de-facto means of creating the service level agreement between the resource consumers and the resource providers. Therefore, standard strategic negotiation models and negotiation protocols have to be designed in order to create and manage agreements. There can be different kinds of negotiation models available which can be applied in different application scenarios. In practice, each such negotiation pattern may be preceded by a discovery phase in which a service publishes or advertises what type of SLAs and negotiation patterns it is willing to support [4], and this information is used by a requestor to select a candidate service provider. To propose and evaluate such strategic models is the main focus in this thesis work.

2.2.6 Policy Issues

Policies are important issues which must be observed in the creation process of the service level agreement. Usually the polices and objectives are internal to the resource providers or users. A resource provider may have complex internal policies for deciding access rights, e.g., resources can only be used by certain user groups, and other scheduling and management procedures. Some of the policies of the resource providers, such as pricing models, may be private. An agreement between the resource provider and the user abstracts these internal policy details and makes the user only know the elements of policy that apply to the agreement.

2.2.7 QoS Support from the Local Resource Management Systems

In order to support the SLAs, the local resource management system has to monitor the jobs during the whole life time of the SLAs and guarantee that the created SLAs can be fulfilled and satisfied. Advance reservation is required in order to support the resource provisioning and, thus make sure the QoSs are fulfilled. An advance reservation is a possibly limited or restricted delegation of a particular resource capability over a defined time interval, obtained by the requester from the resource owner through a negotiation process. Also considering the complex requirements of the job users and the various types of the resources, not only the computing resources, but also the network and storage can be provisioned.

2.2.8 Accounting and Billing

The resource usage data for the job execution should be monitored, collected and accounted during the job execution process. The accounting information is quite important for supporting the agreement based resource management system. The accounting information can be collected via the local resource management systems or some MetaScheduling service. The accounting and billing service will usually connect with the job monitoring service which monitors whether the created agreement terms are qualified. The resource provider/user has to pay some kind of penalty if either of them violates the created service level agreement terms. The users should have access to the accounting and billing information and check whether the information is correct. Billing service is in accordance with the created agreement terms and the penalty policy.

2.2.9 Fault Tolerance

Moreover, the Grid computing infrastructure has to be in high availability by the fault tolerance mechanisms which are needed to monitor, detect the possible fault in the job execution process. Automatic fault tolerance techniques have to be provided, for instance, check point recovery, re-starting the job, etc. Remote backup and simplifying or automating recovery procedures is required.

2.3 Related Work

Resource management and scheduling is a quite important component for the Grid infrastructure. There are extensive research efforts on this area. Since our current research work is mainly focused on agreement based resource management, the related works in this area are briefly reviewed. In the following, the research efforts for automatic negotiation in agent theory are briefly reviewed. Also the Grid resource management approaches, e.g., economic method and matchmaking approach, are discussed. Some existing negotiation protocols, e.g., different auction models, Contract Net, WS-Agreement, are also evaluated.

2.3.1 Computational Mechanisms Design

The Grid and agent communities both develop concepts and mechanisms for open distributed systems, although from different perspectives. The Grid community has historically focused on "brawn": infrastructure, tools, and applications for reliable and secure resource sharing within dynamic and geographically distributed virtual organizations. In contrast, the agents community has focused on "brain": autonomous problem solvers that can act flexibly in uncertain and dynamic environments [13]. Therefore, in this subsection, the general agent concept and the automatic negotiation research in agent community as well as the computational mechanisms design are discussed.

An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives [20, 21]. There are a number of points about this definition that require elaboration. Agents are [21, 22]:

- clearly identifiable problem solving entities with well-defined boundaries and interfaces;
- situated (embedded) in a particular environment and therefore, they receive inputs related to the state of their environment through sensors and they act on the environment through effectors;
- designed to fulfill a specific purpose and they have particular objectives (goals) to achieve;
- autonomous which means that they have control both over their internal state and over their own behavior;
- capable of exhibiting flexible problem solving behavior in pursuit of their design objectives, that is, they need to be both reactive (able to respond in a timely fashion to changes that occur in their environment) and proactive (able to act in anticipation of future goals).

Automatic negotiation has been a research endeavor in the agent community. Automatic negotiation support can save the labor time of the human-beings, furthermore, the computational agents can be more effective at finding better agreement terms than human beings in strategically and combinatorially complex settings [23]. To this end, the multi-agent based approaches are natural models for complex decentralized systems, like the Grid computing and peer to peer systems.

In the multi-agent strategic negotiation scenarios, the interaction protocol and the strategies of different agents can be designed and defined. Figure 2.1 shows that adopting an agent-oriented approach to system engineering means decomposing the problem into multiple, interacting, autonomous components that have particular objectives to

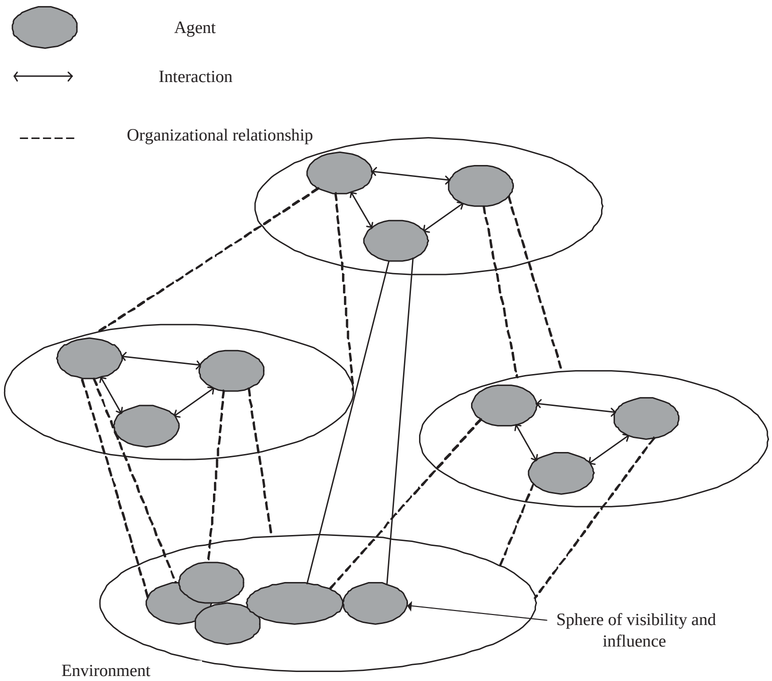

Figure 2.1: Canonical View of a Multiagent System

achieve and are capable of performing particular services. The key abstraction models that define the agent-oriented mindset are agents, interactions and organizations. Finally, explicit structures and mechanisms are often used to describe and manage the complex and changing web of organizational relationships that exist between the agents [13, 24]. In such settings, the agents can cooperate to find a good system wide solution. However, usually this is not feasible because the agents represent distinct stakeholders with potentially conflicting goals that seek to maximize their own gains, e.g., in Grid environments. That is, in the multi-agent environments, there is no central control among them, and usually these agents are selfgoal-oriented without considering the overall social welfare [23]. Consequently, the best a designer can achieve is a noncooperative strategic analysis, in which the designer can impose only the protocol and can not control which strategies the agents adopt. The computational mechanism design is to provide a mathematical framework in which to study protocols that give the agents incentive to act and interact in particular ways and that have useful computational properties [25]. It offers a powerful suite of tools for analyzing, predicting, and controlling the behavior of self-interested agent.

The computational mechanism design has to consider the following characteristics [25]:

- Agents do not have the unbounded computational power that might be required to calculate their preferences for all possible outcomes or calculate equilibrium strategies.
- The agents set are dynamic, the agent can be unavailable at any time.
- Communication cost between different agents must be considered which is not free.
- The system output may not be tractable and obtained using the centralized mechanism.
- Bounded rationality of the agent should be considered.

Some of the various criteria used for judging effectiveness of the negotiation mechanisms are [23]:

- **Social welfare**. Social welfare is the sum of all agents' payoffs or utilities in a given solution. It measures the global good of the agents. It can be used as a criterion for comparing alternative mechanisms by comparing the solutions that the mechanisms lead to.
- **Pareto efficiency**. Pareto efficiency is another solution evaluation criterion that takes a global perspective. Social welfare maximizing solutions are a subset of Pareto efficient ones.
- **Individual rationality**. Participation in a negotiation is individually rational to an agent if the agent's payoff in the negotiated solution is no less than the payoff that the agent would get by not participating in the negotiation.
- **Stability**. Among self interested agents, mechanism should be designed to be stable (non-manipulable). Sometimes, the agents have dominate strategies. However often an agent's best strategy depends on what strategies other agents choose.

In such settings, dominant strategies do not exist and other stability criteria are needed. The most basic one is the Nash equilibrium [26, 27].

- **Computational efficiency**. The mechanisms should be designed so that when agents use them as little computation is needed as possible.

- **Distribution and communication efficiency.** Distributed negotiation protocols should be preferred in order to avoid a single point of failure and a performance bottleneck. At the same time, one would like to minimize the amount of communication that is required to converge on a desirable global solution. In some cases, these two goals conflict.

Negotiation protocol is an important integrated part of the negotiation mechanism. In the next following 3 subsections, the auction protocols, contract net as well as the WS-Agreement protocol are discussed.

2.3.2 Auction Protocols

Auctioning is the method of deciding the value of commodity which has undeterminable price. There are many different auction models proposed and investigated in the economic community. Auctions have one auctioneer and several bidders. Below, four typical kinds of single-sided auction models and the continuous double auction (CDA) are explained.

- **English (first-price open-cry) auction**. In this auction, the auctioneer begins the auction with the reserve price (lowest acceptable price). Each bidder is free to raise his bid. When no bidder is willing to raise anymore, the auction ends and the highest bidder wins the item at the price of his bid. The agent's dominant strategy [23] is to bid a small amount more than the current highest bid and stop when the user's valuation is reached.

- **Dutch auction (descending auction)**. In the Dutch auction, the seller begins the auction with a higher asking price and continuously lowers the price until one of the bidders takes the item at the current price. The Dutch auction is strategically equivalent to the first-price sealed-bid auction. This is because in both games an agent's bid matters only if it is the highest, and no relevant information is revealed during the auction process.

- **Vickery (first price second price) auction**. In a private value Vickrey([23, 28]) auction, the dominant strategy is to bid the user's true valuation. In this context, agents truthfully reveal their preferences which allows efficient decisions to be made.

- **First-price sealed-bid auction (FPSB)**. In this type of auction all bidders simultaneously submit bids so that no bidder knows the bid of any other participant. In the First-price sealed-bid auction each bidder submits one bid without knowing the others' bids. The highest bidder wins the item and pays the amount of his bid. The bidder will bid according to his private value and prior beliefs of others' valuations. In general there is no dominant strategy for bidding in this auction. An agent's best strategy is to bid less than his true valuation, but how much less depends on what the others bid. The agent would want to bid the lowest amount that still wins the auction given that this amount does not exceed his valuation.

- **Continuous double auction (CDA)**. In the continuous double auction, the traders can make offers to buy or sell and to accept other traders' offers at any moment during a trading period [29]. The messages exchanged generally consist of bids (offers to buy) and asks (offers to sell) for single units of the commodity, and acceptances of the current best bid or ask.

Auction models are widely used in the E-Commerce, e.g. Ebay [30]. However, in the auction models shown above, the sellers and the buyers can not easily make the offers and counter offers in an interacting sequential way.

2.3.3 Contract Net Protocol

Contract net protocol is a high-level protocol for communication among the nodes in a distributed problem solver [31, 32]. It is the one of earliest work on the negotiation protocol in the distributed artificial intelligence and the agreement and contracts concept is introduced to the co-operative distributed problem solving. It facilitates distributed control of cooperative task execution. The contract net protocol consists of a collection of nodes, referred to as contract net, where each node in the net may take on the role of a manager, responsible for monitoring the execution and processing the result of a task, or a contractor, responsible for the actual execution of the task. Roles can be adopted dynamically by all nodes at runtime, therefore nodes are not designated a priori as managers or contractors. Typically, a node will take on both roles, often simultaneously for different contracts. That is, the nodes are not statically tied to a control hierarchy. In contract net, a contract is established by a process of local mutual selection based on a two-way transfer of information.

2.3.4 WS-Agreement Protocol

Current Grid research works are towards standardizing resource management functionalities. Here, the Open Grid Services Architecture (OGSA) by the Open Grid Forum (OGF) [33] is a prominent approach to design a core framework for building Grid systems. "WS-Agreement" [34] is a protocol proposed by the Grid resource allocation agreement working group in the OGF. This protocol can be used as a simple negotiation protocol. Web services agreement specification (WS-Agreement) [34] is a web services protocol for establishing agreement between two parties, such as between a service provider and consumer, using an extensible markup language (XML) language for specifying the nature of the agreement, and agreement templates to facilitate discovery of compatible agreement parties. The specification consists of three parts which may be used in a composable manner: a schema for specifying an agreement, a schema for specifying an agreement template, and a set of port types and operations for managing agreement life-cycle, including creation, expiration, and monitoring of agreement states.

The WS-Agreement protocol is dependent on WS-Adressing [35] and Web Services Resource Framework (WSRF [36, 37, 38]), which is proposed by the OGF and some major IT companies including IBM, HP, etc. WSRF is used to enable web services to access state in a consistent and interoperable manner. The WS-Resource approach is adopted to declare and implement the association between a web service and one or more state components. The WS-Resource Framework uses some other Web services specifications, currently e.g. using WS-Adressing [35]. In this approach, the state is modeled as stateful resources and is used to denote the relationship between web services and stateful resources in terms of the implied resource pattern. When a stateful resource participates in the implied resource pattern, it will be denoted as a WS-Resource. WSRF is composed of several specifications [38, 39, 40, 41] and WS-Notification [42, 43, 44, 45] specifications. In WS-Agreement protocol, WS-ResourceProperties and WS-ResourceLifetime are used to represent Agreements as Resources. WS-Agreement is also meant to be composable with other Web services specifications.

According to the WS-Agreement proposal, the conceptual model for the architecture of WS-Agreement based system interfaces has two layers: the *service* layer and the *agreement* layer.

1) The service layer represents the application-specific or domain-specific layer of service being provided. Some domain specific port types can be provided in this layer if the domain-specific resource can be virtualized into web services. Some other domains can not provide web services, but the negotiation can be processed using the web services in which some metadata denotes the qualities of services.

2) The agreement layer provides a Web service-based interface that can be used to represent and monitor agreements with respect to provisioning the services which are implemented in the service layer. The agreement layer has the following port types:

   – An agreement port type has operations to get agreement state and metadata of the agreement such as terms, context, etc. These terms and context are the resource properties of a web service agreement in the WSRF fashion.

   – An agreement factory exposes an operation for creating an agreement out of an input set of terms considering the agreement template and the current resource situation. It returns an EPR (End Point Reference [35]) to an agreement service. The agreement factory also exposes resource properties such as the templates of offers acceptable for creation of an agreement.

In the current WS-Agreement proposal, the negotiation process is a one-shot approach in which negotiation parties can only accept or reject opponent's proposals. This one-shot negotiation process is very time consuming and inefficient since the negotiation opponents have no means to analyze why a proposal is unacceptable, nor in which dimension or direction of the available agreement space a potential solution may exist [21]. To improve the efficiency of the negotiation, the negotiation process should be multi-rounded based. The design of a suitable negotiation protocol is one of the next issues addressed by the Grid resource allocation and agreement protocol (GRAAP) [46] working group. In addition, a common infrastructure for the implementation of this dynamic negotiation process is necessary. It is anticipated that the WS-Agreement and OGSA Basic Execution Services WG (OGSA-BES-WG) [47] work will provide (at least parts) a suitable foundation.

2.3.5 Grid Economics

Economic methods for computational tasks in Grids have been subject of research for some time. Economic method uses the idea of microeconomics theory. In the microeconomics theory, several market models are assumed, for instance, complete competitive market, monopoly market model, oligopoly, etc. The main criteria by which one can distinguish between different market forms are: the number and size of producers and consumers in the market, the type of goods and services being traded, and the degree to which information can flow freely [48]. In an economic-based resource management environment, resource management systems provide mechanisms and tools that allow resource consumers and resource providers to express their requirements and facilitate the realization of their goals. Typically consumers use a utility model to define the requirements and objectives. A brief overview of such models is given below.

Buyya et al. [49, 50] identified the key requirements that an economic-based Grid systems supports and developed a distributed computational economy framework called the GRACE, which is generic enough to accommodate different economic models and maps well onto the architecture of wide area distributed systems. They built a so called Nimrod-G grid resource broker which supports deadline and budget constrained and quality of service requirements-driven application scheduling on world-wide distributed resources. Different scheduling algorithms with four different strategies: cost, time, conservative-time, and cost-time optimizations are adopted in such resource broker.

Ernemann et al. [51, 52] applied the economic models in the Grid scheduling and proposed a scheduling infrastructure that implements a market-approach. Simulations using real workload traces were conducted and the evaluation results showed that economic scheduling algorithms provides average weighted response-times as good or better than a common scheduling algorithm with backfilling. This economic model has the additional advantages of supporting different price models, different optimization objectives, varying access policies, and Quality of Service demands.

Wolski et al. [53] investigated the so called G-commerce computational economies for controlling resource allocation in computational Grid settings. Hypothetical resource consumers (representing users and Grid-aware applications) and resource producers (representing resource owners who "sell" their resources to the Grid) were defined. The efficiency of resource allocation under two different market conditions: commodities markets and auctions were measured. Both market strategies were compared in terms of price stability, market equilibrium, consumer efficiency, and producer efficiency. The results indicated that commodities markets are a better choice for controlling Grid resources than previously defined auction strategies.

2.3.6 Matchmaking

Matchmaking approach is adopted in the Condor project [54]. The matchmaker performs scheduling in a Condor pool, resource requests and offers are described in the Condor classified language and the matchmaker is responsible for finding suitable resources to satisfy the needs of the job users.

This is the approach employed in Condor, which is a high throughout computing (HTC) resource management system for large collections of distributively owned computing resources. Resource requests and offers are described in the Condor classified advertisement (ClassAd) language. The matchmaker performs scheduling in a Condor pool. The matchmaker is responsible for initiating contract between compatible agents. However, Condor is a system-centric Grid resource management system.

2.3.7 Agreement Based Resource Management

To this end, a lot of efforts have been made on the Grid resource management considering the service level agreement (SLA). Czajkowski et al. [4] introduced the concepts of agreement-based resource management in the Grid computing environment and presented a general agreement model. Dumitrescu et al. [55] presented and evaluated a Grid resource usage SLA broker called GRUBER in a real grid, GRID3. Padgett et al. [56] proposed an architecture for specifying, monitoring and validating SLAs for use in Grid environments. Sim [57] reviewed and compared the very few existing research initiatives on applying bargaining as a mechanism for managing Grid resources. Sandholm [58] proposed a service level agreement and agent-based architecture, in which the issues involved in managing complex policies of multiple stakeholders in the large-scale, dynamic, and heterogeneous Grid are discussed and addressed.

Although there are a lot of efforts on the agreement based resource management for Grid computing, none of the presented models include general negotiation schemes for independent Grid schedulers. Also, strategies for conducting the negotiation between the participating parties are not yet well understood. There have been several efforts as introduced in the community of economics which are not yet well analyzed to the Grid scenario. Here, such strategic negotiation models as well as additional work in regards of the influences of the strategies are required.

2.3.8 Resource Co-Allocation in Grid Computing

Resource co-allocation as one of the most challenging problems in the Grid computing has been investigated for quite some time. In the Condor project [16], gang matchmaking scheme is used, which extends the bilateral matchmaking to the multilateral matchmaking model in order to support the co-allocation. Yahyapour et al. [59, 60] evaluated multi-site scheduling (co-allocation) in Grid environments using typical scheduling objectives, in which the advantages of multi-site job execution were identified considering overhead for data and communication costs. Epema et al. [61] discussed the co-allocation problem in the Grid and built a grid scheduler called KOALA which supports the co-allocation. The co-allocation problem for computational Grids has been defined in [62], in which some co-allocation mechanisms are proposed. Kuo et al. [63] proposed an advance reservation and co-allocation protocol for grid computing; but negotiation and scheduling issues are not dealt with. Buyya et al. [49] considered economic scheduling for Grid resources in the NIMROD-G and GridBus [50] projects. However, optimization was limited to specific objectives. In the VIOLA project, a web service based meta-scheduling service which allows to negotiate a common time slot with local resource management systems to enable the execution of a distributed workflow [64] is presented and evaluated, in which a negotiation protocol is proposed that the metascheduling service uses to negotiate the allocation of resources with the local scheduling systems. Roeblitz et al. [65] presented an architectural framework for specifying and processing co-reservations in Grid environment. The virtual resource concept in this framework allows an easy integration of multiple physical resources and it provides the means for introducing advanced usage scenarios like multi-site jobs with temporal and spatial dependencies. In distributed systems creating consensus among parties can be a complex issue. There have been several consensus algorithms proposed [66, 67] which can be used in building resource management systems to support the co-allocation.

However, to this end, there is no common negotiation model proposed and evaluated for the co-allocation in the Grid environment.

2.4 Research Focus

As introduced before, agreement based resource management approach is adopted in the current research work. Negotiation is the de facto means of creating such bi-lateral service level agreements (SLAs) between Grid parties. Such negotiation processes should be automated with no or minimal human interaction, considering the potential scale of Grid systems and the amount of necessary transactions. Therefore, strategic negotiation models play important roles. The Grid computing infrastructure is neither the complete competitive market nor the monopoly market. Therefore the strategic bargaining theory [68] will play a role.

Considering there are many different negotiation scenarios, interactions between agents can be classified into three types [69] in terms of the number of agents negotiation scenarios: *one to one*, *many to one* (or one to many), and *many to many*.

One to one negotiation is that one agent is negotiating with exactly one other agent which is important for both theoretical and practical reasons. Theoretically, it is important because of the technical difficulties that this apparently simple setting provides; for example most games that result from one-to-one negotiation can be proven to have multiple equilibria [70], and a naive application of game-theoretical tools is therefore not possible. Practically, it is important because of the emerging role of one-to-one relations associated with business-to-business e-commerce scenarios.

Many-to-one negotiation where many agents negotiate with just one agent is the standard setting of auctions, which have been popular on the Internet for some time now, e.g. Ebay [30]. In this setting, one agent plays the role of the seller, while many play the role of the buyers. Other settings are possible such as having many sellers and one buyer.

Finally, many-to-many negotiation (where many agents negotiate with many other agents) constitutes the most complex scenario. The continuous double auction is the most complex of these scenarios which has introduced before [29].

In the current research, the focus is only on the negotiation process in which a user negotiates with a set of resource providers. In our model the user, or more precisely some meta-scheduling agent or job broker on his behalf, will contact different resource providers, negotiate with several of them and make a decision to commit to a particular agreement with one resource provider. This is considered as the *one to many* negotiation type [69]. Usually, this negotiation type can be treated as "reversed auctioning" [71]. However, there are some drawbacks of using auction mechanisms, for instance, there is no flexible way of exercising different strategies with different negotiation opponents. Moreover, as introduced before, auctions do not support bidirectional offers with counter offers between parties. Therefore the multi-bilateral negotiation models are adopted in the current research work.

The bilateral negotiation model is the basis of multi-bilateral negotiation model, so bilateral negotiation model will be firstly introduced.

In the bilateral negotiation models, the two agents involved are cooperative when they have common interests and in conflict situation because they have different objects in the negotiation process. Therefore, we can divide the bilateral negotiation into cooperative bargaining type and non-cooperative bargaining [72] type. The bargaining is a type of game, which is originated with the work of von Neumann [73] and John Nash [26, 27]. Cooperative game theory is the complete information game, in which the preference information of a negotiation party is known to all other negotiation parties. It abstracts away from specific rules of a game and is mainly concerned with finding a set of possible outcomes. The solution is required to satisfy certain plausible properties, such as the stability or fairness, which are called axioms. Non-cooperative game theory, on the other hand, is concerned with specific games with a well-defined set of rules and game strategies, which are known by the players before the bargaining or negotiation begins. In the non-cooperative game theory, there can be complete information games and incomplete information games. The incomplete information game can be further divided into one-sided incomplete information game and two-sided incomplete information game. If only one side has its private negotiation information, the game is called one-sided incomplete game. If both sides have their private negotiation information, the game is called two-sided incomplete information game.

In the Grid environment, the users can not know the complete information of the resource providers (the current resource status, the negotiation preferences of the providers, the cost model, etc). The resource providers can not have all of the related information of the users (different policies and objectives, maximum acceptable price, etc.). The negotiation in the Grid computing is a two-sided incomplete bargaining case. In the following, the strategic bilateral negotiation model is explained.

2.4.1 Strategic Bilateral Negotiation Model

There are three parts in the bilateral negotiation model that have to be considered [21]: 1) the negotiation protocol, 2) the negotiation issues/objects, 3) the negotiation strategy that is applied during the negotiation process.

- **Negotiation Protocols**: negotiation protocols are the set of rules that govern the interaction, which define the possible types of participants (e.g. the negotiators and any relevant third parties), the negotiation states (e.g. accepting bids, negotiation closed), the events that cause negotiation states to change (e.g. no more bidders, bid accepted) and the valid actions of the participants in particular states (e.g. which messages can be sent by whom, to whom, at what stage). It also defines whether the negotiation is one-shot or multi-rounded based.

- **Negotiation Objects/Issues**: The range of issues over which agreement must be reached. There can be many different negotiation scenarios. The object may contain a single issue (such as price), while on the other hand it may cover hundreds of issues (related to price, quality, timings, penalties, terms and conditions, etc.). Orthogonal to the agreement structure, and determined by the negotiation protocol, is the issue of the types of operation that can be performed on agreements. In the simplest case, the structure and the contents of the agreement are fixed and participants can either accept or reject it (i.e. take it or leave it offer). At the next level, participants have the flexibility to change the values of the issues in the negotiation object (i.e. they can make counter-proposals to ensure the agreement better fits their negotiation objectives). Finally, participants might be allowed to dynamically alter (by adding or removing issues) the structure of the negotiation object.

In general, for the multi-issued based negotiation scenarios, there are a number of different procedures that can be used for determining the negotiation process [74]; the three main ones being the

- **package deal procedure** in which all the issues are bundled and discussed together;
- the **simultaneous procedure** in which the issues are discussed simultaneously but independently of each other;
- and the **sequential procedure** in which the issues are discussed one after another.

- **Negotiation Strategies**: the decision making tools the participants employ to act in line with the negotiation protocol in order to achieve their objectives. The sophistication of the model, as well as the range of decisions that have to be made, are influenced by the protocol in place, by the nature of the negotiation object, and by the range of operations that can be performed on it.

In the Grid computing scenarios, there are many negotiation issues involved, e.g., cost, response time of the job, waiting time of the job etc. In our research work, the multi-issue negotiation is handled using the package deal procedure.

2.4.2 Negotiation Models and Strategies Research for Bilateral Negotiation

For the two-sided incomplete information bilateral negotiation, there are many negotiation models and strategies proposed in the agent community. Below are some of these models and strategies. Some of these models or strategies are leveraged in our research work for Grid scheduling.

Faratin et al. [75] devised a negotiation model that defines a range of strategies and tactics for generating proposals based on time, resource, and behaviors of negotiators. The time-based strategy for the bilateral negotiation model is adopted in our current research work which will be discussed and evaluated in more details in Chapter 3. Sim et al. [76] proposed a market-driven model for designing negotiation agents that make adjustable rates of concession by reacting to some essential market situations that could change over time. The market-driven strategies were further improved in Sim and Wang [77] with a set of fuzzy rules to enhance the flexibility of negotiation agents. The market situations include trading opportunities, competition, remaining trading time, and eagerness. In their model the number of trading opportunities influences the aggregated probability of conflict, which determines the probability of completing a deal in the current negotiation cycle.

In agent research, learning is an important technique to deal with an environment of which complete knowledge is not known. In bargaining games [78], learning is potentially important in the following two aspects. First, a bargaining agent can adjust its bargaining strategy in order to achieve better deals, which is based on its experiences in previous bargaining games. Second, it is useful to update the belief of the other parties' types or strategies in the negotiation process by observing the behaviors of the other parties, and then to adjust one's strategy accordingly. Gerding et al. [78] reviewed several learning techniques including decision trees, Q-learning, evolutionary algorithms and Bayesian beliefs. Emphasis is given to applying evolutionary approaches and Bayesian beliefs to learn effective strategies or useful information in negotiations. Zeng et al. [79] presented a sequential negotiation model and address multi-agent learning issues by explicitly modelling beliefs about the negotiation environment and the participating agents under a probabilistic framework using a Bayesian learning representation and updating mechanisms. Excelente-Toledo et al. [80] examined the potential and the impact of introducing learning capabilities into autonomous agents that make decisions at run-time about which mechanism to exploit in order to coordinate their activities. Narayanan et al. [81] adopted the Markov chain framework to model bilateral negotiations among agents in dynamic environments and use Bayesian learning to enable them to learn an optimal strategy in incomplete information settings. Specifically, an agent learns the optimal strategy to play against an opponent whose strategy varies with time, assuming no prior information about its negotiation parameters.

There are also some research works focused on the multi-bilateral negotiation which is also related to our current research work. Nguyen and Jennings [82, 83] presented a heuristic model that enables an agent to participate in multiple, concurrent bi-lateral negotiations in which there is incomplete information and time deadlines. In this model the buyer that has outside options can accept an offer from a seller, with the agreement binding only on the seller but not on the buyer. In other words, the buyer can decline the agreement that is not finalized if she finds a better deal later. This protocol is extremely buyer-biased as the buyer is guaranteed the best offer she can find from all different threads. In reality the buyer is usually not a monopoly player. Sellers may also have outside options and can opt out or withdraw an offer before the buyer finalizes the decision. Another key problem in the concurrent bilateral model is managing commitments since an agent may want to make intermediate deals (so that it has a definite agreement) with other agents before it gets to finalize a deal at the end of the encounter. Nguyen and Jennings [84, 85] extended this concurrent multi-lateral model with the more flexible commitment model in which the agents can reason about in order to determine when to commit and to decommit. This model is based on the leveled commitment protocol [86, 87] in which both the users and buyers can decommit the deals simply by paying an amount of penalty. Li et al. [88] presented a model for bilateral negotiations that considers the uncertain and dynamic outside options. Outside options affect the negotiation strategies via their impact on the reservation price. The model is composed of three modules: single-threaded negotiations, synchronized multi-threaded negotiations, and dynamic multi-threaded negotiations. The single-threaded negotiation model provides negotiation strategies without specifically considering outside options. The model of synchronized multi-threaded negotiations builds on the single-threaded negotiation model and considers the presence of concurrently existing outside options. The model of dynamic multi-threaded negotiations expands the synchronized multithreaded model by considering uncertain outside options that may come dynamically in the future.

2.4.3 Evaluation of the Negotiation Results

In order to evaluate the negotiation results, criteria should be defined. It can be defined from the perspective of either the individual or the global aspects. In the cooperative game theory, the Nash solution is the most popular solution point to the bargaining problem [26, 27]. The other is the reference point. This is observed in experimental bargaining problems where a prominent outcome is used by negotiators to anchor a point in the set of outcomes [89]. The negotiators can then use this reference point as point of improvement to the final point. This point can be used either as a commonly agreed on starting-point, a credible final point, or simply a focal point [90, 91]. In multi-issue negotiations, the mid point of each issue of both agents' reservation can serve as such a reference point, from which negotiators may attempt to jointly improve [89, 92]. In the extensive games, the Nash equilibrium is unsatisfactory since it ignores the sequential structure of the decision problems [70]; therefore, the wrong Nash equilibria will be obtained [93]. In the extensive games with complete information, the subgame perfect equilibrium exists. A strategy profile is a subgame perfect equilibrium in a model of alternating offers (extensive games) if the strategy profile induced in every subgame is a Nash equilibrium of that subgame [94]. While in the non-cooperative two-sided incomplete games (incomplete bilateral negotiations), both agents may not know the type of their opponents and their presences and private information. An agent's negotiation strategy is any function of the history of the negotiations to its next move. In this kind of the negotiation, both negotiation parties have incomplete information about opponents' information, therefore, the agents' strategies should take their belief into consideration. The sequential equilibrium exists [68, 70]. Sequential equilibrium is a refinement of Nash Equilibrium for extensive-form games due to Kreps and Wilson [95]. In the sequential equilibrium, the main point for the agent is how to use its beliefs in the negotiation and how to update its beliefs according to the information it collects in the negotiation process, and also how an agent influences its opponents' beliefs. Sequential equilibrium requires that in each time period any agent's strategy will be optimal given its opponents' strategies, the history up to the given time period, and its beliefs. The notion of sequential equilibrium requires the specification of two elements: the profile of the strategies and the beliefs of the agents.

2.4.3.1 Discrete Event Based Simulation

Theoretical analysis gives some hints on evaluating the negotiation models according to some specified criteria and assumptions. However, in Grid environment, even the local resource management systems are hardly evaluated using the theoretical approaches as the theoretical worst case analysis is only of limited help as typical workloads on production environments rarely exhibit the specific structure that will create a really bad case [52]. In order to evaluate the negotiation models, we have to consider the local resource management system as well. Also the proposed negotiation strategies in our research work are heuristic in nature, there are many parameters involved in a broad range of negotiation situations. Therefore, in our research work, discrete event based simulation approach using workloads is used to evaluate the proposed model, which will explained in detail in the following chapters.

In the next following chapters, the strategic models for Grid scheduling including the negotiation protocol, utility functions as well as the negotiation strategies are proposed and evaluated.

Chapter 3

A Strategic Negotiation Model Supporting a Simple Grid Scheduling Scenario

Prior to service usage the parties have to negotiate towards service level agreements (SLA) that define what kind of services will be provided and what the obligations of the user will be. Therefore, strategic negotiation models and strategies play important roles in the negotiation process. In this chapter, a negotiation model for Grid computing is proposed and evaluated.

3.1 Strategic Negotiation Model

3.1.1 Bilateral Negotiation Model

There are three parts in the bilateral negotiation model that have to be considered [94]: 1) the negotiation protocol, 2) the used utility/preference functions for the negotiating parties, and 3) the negotiation strategy that is applied during the negotiation process.

3.1.1.1 Negotiation Protocol

In our approach, we adopted and modified Rubinstein's sequential alternating offer protocol for Grids, see [96]. In Rubinstein's alternating offers bilateral negotiation protocol, the bargaining procedure is as follows: The players can take actions only at certain times in the (infinite) set $T = \{1, 2, 3, ...t\}$. In each period $t \in T$, one of the players, say i, proposes an agreement, and the other player j either accepts the offer or rejects it. If the offer is accepted, then the bargaining ends, and the agreement is implemented. If the offer is rejected, then the process passes to period $t+1$; in this period player j proposes an agreement, which player i may accept or reject. The negotiation process will go on in this way.

In the Grid resource management scenario, time plays an important role as every negotiation party has only limited negotiation time available. Therefore, the number of the negotiation rounds is limited. In our scenario, the above time set T is finite. In the negotiation process, when either one negotiation side times out or an agreement is created, the negotiation process will end. An offer is assumed to be valid until a counter offer is received. Therefore the consistent state problem between the negotiation parities can be avoided.

3.1.1.2 Utility Functions

As mentioned before, we support utility functions to express the objectives of the users; preference relationships are used to indicate the preferences of resource providers. Usually, the objectives of the user require minimizing the job waiting time or to get cheaper resources; on the other side, the resource providers expect to gain higher profit and higher utilization. However, the real weighting of the utility factors depends on the individual user or resource provider. In real Grid systems, the users or resource providers may have many different negotiation objectives, that are interdependent and should be dealt simultaneously (that is, we deal with the multi-issues negotiation as the package deal type) which yields to a multi-criteria optimization problem [97].

In the following we consider as first examples the expected waiting time of the jobs and the expected cost per cpu time as the negotiation issues. However, the model can be applied and extended to other criteria as well. In this model, $U_{price}(P_c^t)$ (E.q.1) is the job's utility function of the price and $U_{time}(T_c^t)$ (E.q.2) is job's utility function of the waiting time.

$$U_{price}(P_c^t) = \frac{P_c^{max} - P_c^t}{P_c^{max} - P_c^{min}} \tag{3.1}$$

$$U_{time}(T_c^t) = \frac{T_c^{max} - T_c^t}{T_c^{max} - T_c^{min}} \tag{3.2}$$

The variables are explained as follows: W_{price} is the weight of the price utility. W_{time} is the weight of the time utility. P_c^{max} (P_c^{min}) is the maximum (minimum) acceptable price of the user offered by the negotiation opponent at the time t. T_c^{max} (T_c^{min}) is the maximum (minimum) acceptable waiting time of the user.

This leads to the following aggregate utility function of the user

$$U_{job} = W_{price} * U_{price} + W_{time} * U_{time} \tag{3.3}$$

Because the negotiation time in this scenario is usually short, the utilities in this scenario are not discounted as negotiation time goes on. The weights of different negotiation issues are normalized, so we assume that $\sum_{j=1}^{n} w_j = 1$ if the number of the negotiation issues is n. In the negotiation process, an agent can change its preference for an issue by changing the weight associated to that issue. Different agents can have quite different preferences over different issues.

For the resource providers, there are also two corresponding negotiation issues which are: the expected waiting time of the job $T_s^t(Job)$, and the expected price $P_s^t(Job)$. The expected waiting time for the newly incoming job can be obtained from the current resource status and the future schedule plan considering the created agreements which have to be fulfilled. The expected price will be obtained via the negotiation process.

The *zone of possible agreement* shown in Figure 3.1 denotes the overlap in the negotiation issues between the participating parties [89]. We assume that the user is the buyer and the resource provider is the seller. Note that, the resource provider will prefer a higher price while the user always prefer a lower price, it is best that the maximum price of the resource provider is infinity and the minimum price of the user is 0. But considering the reality of the Grid computing and for the facility of initializing the negotiation, we assume that there will be maximum prices (not infinity) for the resource providers and minimum prices for the users. If there is no zone of possible agreement, an agreement can not be achieved. For the negotiation issue j, we assume that the reservation value (maximum acceptable value) of the user (buyer) is C_{max}^j and the minimum value of the user (buyer) is C_{min}^j; the reservation value (the minimum acceptable value) of the resource provider (seller) is S_{min}^j and the maximum value of the resource provider (seller) is S_{max}^j. Therefore, the acceptable value range of the user (negotiation zone of the buyer) is $[C_{min}^j, C_{max}^j]$; while the acceptable value range of the resource provider (negotiation zone of the seller) is $[S_{min}^j, S_{max}^j]$. If $C_{max}^j > S_{min}^j$, then the agreement zone exists.

In the negotiation process, both the objectives of the users and resource providers are to get more of the so called negotiation surplus which is shown in the figure.

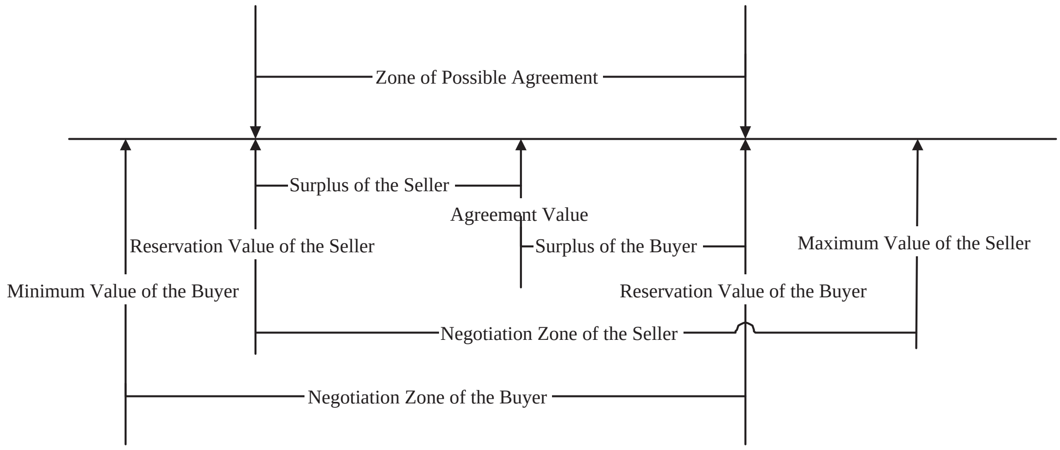

Figure 3.1: Zone of Possible Agreement

In the negotiation process, our negotiating parties act rationally. Disagreement is treated as the worst outcome, therefore the negotiation party always avoids opting out of the negotiation process. One of the principles of good-faith bargaining is that once a concession is made, it is usually not easily reversed [89]. On the basis of the initial values, successive offers by sellers are monotonically decreasing while successive offers by the buyers are monotonically increasing. It is important that the negotiating parties provide suitable initial values for the negotiation issues.

In the negotiation model, the negotiation parties can not know the opponents' private reservation information and their preferences/utility functions. Without this restriction, the parties could exploit the condition of the corresponding negotiating partners. That means, a negotiation scenario with *incomplete information* is considered. In the negotiation process, the negotiation parties should make reasonable reservation values of different negotiation issues in order to make sure that it is possible to create agreements.

3.1.1.3 Negotiation Strategies

In the strategic negotiation model there are no rules that bind the negotiation parties to any specific strategy. The essence of the negotiation strategy for the negotiation party is to create suitable offers in its acceptable value range of specific negotiation issue in order to create the agreement and make its utility as much as possible at the same time. As shown before, the negotiation parties do not know the reservation values and the utility functions/preferences of the opponents in our scenario. Therefore, *heuristic based negotiation strategies* are adopted in the current research work. The negotiation process in the Grid computing domain is time-limited, the strategies of the negotiation parties are considered to change dynamically based on the remaining available negotiation time. Typically, a user will not negotiate and wait for the negotiation result for a long time, if he/she has a very urgent job needed to be executed. To this end, we limit our scope on *time dependent negotiation strategies* [75]. However, note that there are also other negotiation strategies available which are based on other assumptions, for example, if there are many resources available for a job, then the job user may become very tough during the negotiation process.

We assume that V_j is the utility function of the negotiation party which associates with the negotiation issue j and the $x^t_{a\rightarrow b}[t]$ is the offer provided by one party (denoted by a) to another negotiation party (denoted by b) at time t where t denotes the current time instant in the negotiation time with $0 \leq t \leq t^a_{max}$, and t^a_{max} is the deadline of the negotiation party a for the completion of negotiation.

If V_j is decreasing:

$$x^t_{a\rightarrow b}[t] = min^a_j + \alpha^a_j(t)(max^a_j - min^a_j), \tag{3.4}$$

if V_j is increasing:

$$x^t_{a\rightarrow b}[t] = min^a_j + (1 - \alpha^a_j(t))(max^a_j - min^a_j), \tag{3.5}$$

Equations 5.5 and 5.6 represent the job user's strategy and the resource provider's strategy respectively. There are many ways of defining the function for $\alpha^a_j(t)$. However, the functions must make sure that $0 \leq \alpha^a_j(t) \leq 1$, $\alpha^a_j(0) = k^a_j$ and $\alpha^a_j(t^a_{max}) = 1$. For the initial bargaining value k^a_j is used, for which the following relation holds $0 \leq k^a_j \leq 1$.

We use the following function for $\alpha^a_j(t)$:

$$\alpha^a_j(t) = k^a_j + (1 - k^a_j)(\frac{t}{t^a_{max}})^{1/\beta}, \tag{3.6}$$

where the parameter β is the degree of convexity that determines the type of the negotiation party in the time dependent strategy. Different β values yield different negotiation strategies.

There are three typical strategies for different negotiation parties [75]. When $0 < \beta < 1$, the negotiator will be tough (Boulware [89]), which means that he will maintain the offered value longer until the time is almost exhausted. Close to the deadline he will concede up to the reservation value. In contrast, for $\beta > 1$ the negotiator will be the type of Conceder [92] and will concede to its reservation value very quickly at the beginning of the negotiation, while its concession rate becomes flattened as the time limits approached. For $\beta = 1$, the negotiator will linearly concede to its reservation value. In Figure 3.2 the value of the $\alpha(t)$ is shown with respect to different β values.

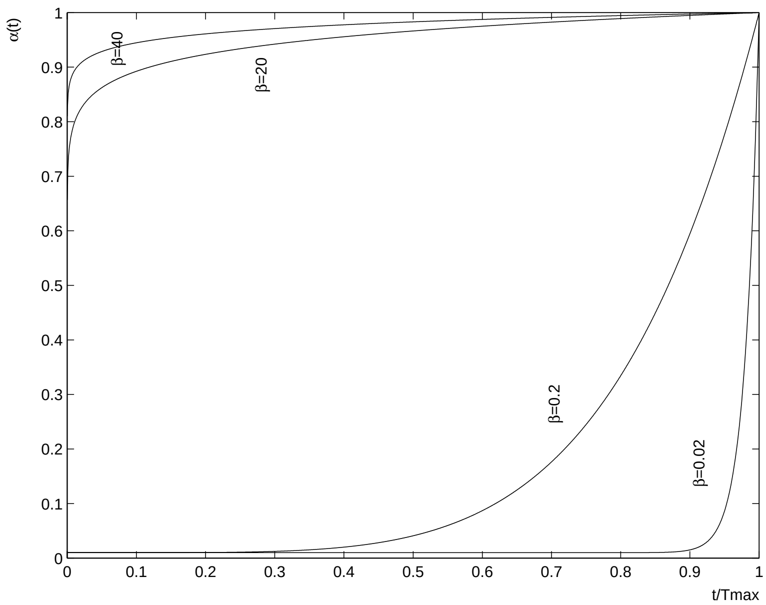

Figure 3.2: Polynomial Functions for the Computation of $\alpha(t)$

3.1.2 Concurrent Bilateral Negotiation Model

As introduced in Chapter 2, in the Grid environment, it is assumed that after a resource discovery phase there are a number of available resources which are capable of fulfilling the constraints of the job. These constraints include, e.g., the required number of CPU nodes, the needed memory capacity. The user or a corresponding scheduling component will contact different resource providers and initiate the negotiation process for the actual resource allocation.

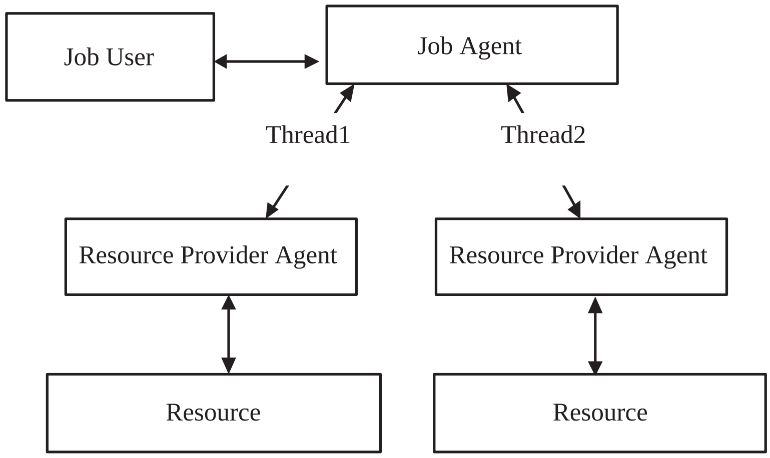

Figure 3.3: Concurrent Negotiation Threads

As shown in Figure 3.3, the job agent acts on behalf of the user and negotiates with the resource agents which act on behalf of the resource providers. We just assume that the negotiation process is started by the user, more precisely by the job agent, who contacts different resource providers and begins the negotiation process. The figure shows an example where two concurrent negotiation threads are running. The job agent and the resource agent interact with each other and try to create an agreement.

In the concurrent negotiation threads in which the same user is involved, the reservation values of the negotiation issues and preferences of the user are the same. However, the user may adopt different strategies with respect to different negotiation opponents. Furthermore, they might change the negotiation strategies during the negotiation process according to the types of opponents and their behaviors. This means that in this process the negotiation agent might change the negotiation threads because these negotiation threads are progressed concurrently, it is very difficult to predict whether the user might achieve a better offer from another negotiation thread if there is already a suitable offer found that could be committed to an agreement. In this chapter, we assume that once there is an agreement available, the agreement is made. This limitation will be extended in the next chapters.

3.2 Evaluation

As introduced in Chapter 2, there are several criteria to judge the negotiation models as well as the negotiation strategies. In the following, the criteria to evaluate the negotiation model for Grid scheduling are proposed.

3.2.1 Evaluation Criteria

Without a reference benchmark for negotiation-based Grid scenarios, it is difficult to compare and analyze the quantitative and qualitative output of such a scheduling model. We use the following criteria for evaluation:

- Comparison between the negotiation result and the reference point [89], which is the middle of the agreement zone of user and resource provider: $[C_j^{max}, S_j^{min}]$. The reference point is computed by the following function:

$$U_j^{ref} = \frac{C_j^{max} + S_j^{min}}{2} \tag{3.7}$$

  For instance, we compare the difference between the agreement price (AP) and the reference price (RP).

- The rate of successfully created agreements for all jobs which is one of the criteria to evaluate whether the negotiation model is applicable in Grid infrastructure or not.

- The negotiation overhead to create the agreement measured by the time taken to create the agreement. In our case, we use the final negotiation rounds which represent the required number of messages exchanged. The actual network overhead will depend on the actual network speed for this message exchange.

- In the Grid computing environment, the users will be interested in the job response time and the waiting time; while for the resource providers the utilization rate of the resource and the profit will probably be the main objectives. We compare these criteria to get some feedback about the feasibility of the negotiation model. In more details, we compare the average weighted response time (AWRT), the average weighted waiting time (AWWT). For the weight in AWRT and AWWT we used the job resource consumption [98]. This weight prevents any favor of jobs with higher or lower resource consumption over each other.

For a first analysis of the approach we use discrete event based simulation. The proposed negotiation model including the negotiation strategies can be evaluated using the real traces or some workload models in the ideal cases. However, currently, there is no real data or general workloads from the Grid computing environments that include suitable information for negotiation models (cost, deadline, etc.). To this end, high performance computing is one of the major applications in the Grid computing infrastructure. There are real workloads available [99] from several real installations. Also there are a lot of works on the workload modeling [100, 101] for parallel computing. Therefore, we use high performance computing workload traces from actual machine installations. However, negotiation information (e.g., the cost information, the acceptable maximum waiting time for the job) is not included in this data as none of these real installations supported negotiation models. To this end, the missing information can only be modeled based on first assumptions. Here we use the simple uniform distribution to create the needed negotiation simulation data. Note, that the presented results may vary for practical implementations with different workloads.

In the following the simulation configuration is described and the simulation results are analyzed.

3.2.2 Simulation Configuration

In the simulation, we investigated different negotiation parameters which possibly have some kind of influence in the negotiation result. As shown in Figure 3.2, if the β value is in the range of $[0.02, 0.2]$, the negotiation party will behave quite tough; while if the β value is in the range of $[20, 40]$, the negotiation party will concede very quickly. We will use these value range areas to do the simulation. In the beginning of the negotiation, the negotiation parties will always make the offers which are most favorable to themselves, at the first assumption, we assume that the initial values of k_j of all the negotiation parties are 0, therefore the job agent will begin the negotiation from its lowest bidding value. We assume that the negotiation interval between every negotiation round is one second, in which time negotiation parties decide to accept the agreement or produce new offers and transfer them to the remote negotiation opponents. In the following we describe the modeling of the user and the resource providers.

3.2.2.1 User Model

In our simulation we consider parallel batch jobs in an online scenario. That means, jobs are submitted from users over time and are not known in advance. We assume that users behave quite differently in the negotiation process. Some concede easily while others behave toughly in the negotiation process. Also the objectives and preferences of different users are quite different, for instance, some of them will prefer time optimization, while others prefer cost optimization. For our simulation, we just assume that there are two different kinds of user objectives: time-optimization and cost-optimization. Below are the parameters of the user modeling which have been applied for the simulation. As introduced before, there are no traces about the bidding information from the real installations as there are no Grid systems which are running using the same negotiation models proposed in the current research work. The needed bidding data can only be generated according to our first assumptions. We assume that the maximum acceptable negotiation span for the user is 30 seconds in which the user will get the negotiation result because usually the user is not willing to wait for a long time to do the negotiation in Grid environment. The maximum acceptable waiting time for the user is 36000 seconds (10 hours). The information of the negotiation span, maximum price, as well as the acceptable waiting time can be easily transformed from one scale into another. The needed simulation data is generated using the simple uniformly distribution. The weights of waiting time and price for the time-optimization and cost-optimization can also be very different from users to users, here we just assume that they are same for the same type of users.

- Negotiation span is uniformly distributed in $[0, 30]$ seconds.
- Maximum price of the different job user is uniformly distributed in $[4.0, 9.0]$.
- Acceptable waiting time for the job users is uniformly distributed in $[0, 36000]$ seconds.
- For the tough negotiator, β value is uniformly distributed in $[0.02, 0.2]$.
- For the conceder negotiator, β value is uniformly distributed in $[20, 40]$.
- Weights of waiting time and price for the time-optimization are 0.8 and 0.2, while the weights of the time and price for the cost-optimization are 0.2 and 0.8.

3.2.2.2 Resource Provider Model

As introduced before in Chapter 1, currently there are many local resource management systems available. Usually First come first serve (FCFS) scheduling algorithms are used in these production environments. Therefore, we use the FCFS scheduling strategy with EASY backfilling [102]. There is no preemption allowed, which means that once a job is started, it will run to completion. In this evaluation we do not yet consider the co-allocation and combination of different agreements from different providers. For the moment, the resources are considered to be homogeneous only differing in the number of available CPU nodes at each site. Different resource providers have different policies, for example, their pricing and negotiation strategies can be different. We assume that job users will contact resource providers which can fulfill their hard constraints. The simulated Grid configurations for the resource providers are consistent with the actual configurations of the systems from which the real traces originated. The negotiation results will be variable with respect to different workloads and resource configurations. The actual quality will have to be verified better workload models and in real implementations. In the current work, we use traces from the Cornell Theory Center [99] because the scheduling algorithms on this machine was performed by EASY algorithms. Of the 512 nodes in the system, 430 are dedicated to running batch jobs; the remainder of the nodes are used for interactive jobs, I/O nodes, special projects, and system testing. The log pertains to the batch partition. The CTC SP2 is heterogeneous in the sense that not all 512 nodes are identical. However, for the simple case, in our current research work, we assume that all of the 512 nodes are dedicated to running batch jobs and homogenous. In our simulation we assumed a Grid scenario with 6 different machines and therefore 6 resource providers. However, to stay consistent with the available workload from the CTC traces, the number of nodes for all simulated machines is also 512 nodes. The number of nodes on each machine is given below, Note also that there are a lot of means of dividing the total nodes into different machines [52, 103] and the different machine configurations have great influences for the scheduling results. But the emphasis of our current research work is on the negotiation model and negotiation strategies, therefore, here we only show one case. The needed negotiation information (maximum prices, minimum prices, negotiation deadlines, etc.)

is also based on our first assumptions. The following list shows negotiation parameters for each resource provider in this scenario.

- The numbers of the CPU nodes are {`384, 64, 16, 16, 16, 16`}.
- Their different maximum prices per CPU time are {`8.2, 8.0, 7.5, 7.6, 7.4, 7.5`}.
- Their different minimum prices per CPU time are {`2.4, 2.3, 2.0, 1.95, 1.90, 1.80`}.
- Negotiation deadlines of different resource providers are all 30 seconds, which means that usually the resource provider will not opt out of the negotiation once the negotiation thread is created.
- For the conceding negotiator, β value is {`32, 35, 34, 38, 40, 40`}.
- For the tough negotiator, β value is {`0.03, 0.05, 0.04, 0.10, 0.05, 0.06`}

3.2.3 Simulation Results and Comparison

In the following we provide some first simulation results which give some information about the performance of the model. We used the first 5000 jobs from the CTC workload traces [99] to do our simulation. As mentioned before, the negotiation parties use different negotiation strategies and they have different reservation values and utility functions/preferences. In the figures we use the following abbreviations: T, L, C denote the tough, linear, and conceding strategies respectively. T-T means both parties act tough, T-C means that the job users are tough, while the resource providers are conceding. We compared four typical different scenarios for our simulations: L-L, C-C, T-C, T-T. Every simulation scenario is represented by every group bar as shown in every result figure. The simulation results are shown from Figure 3.4 to Figure 3.9. Note, that, in every group bar of the figure except the first figure, there are six bars which represent the result of resource one to resource six separately.

In Figure 3.4 and Figure 3.5, we can see that the C-C scenario provides the highest number of successfully created agreements, as well as the highest resource utilization. While in the T-T scenario, the number of the created agreement is lowest and the utilization rate of the resources is also lowest compared to other simulation cases.

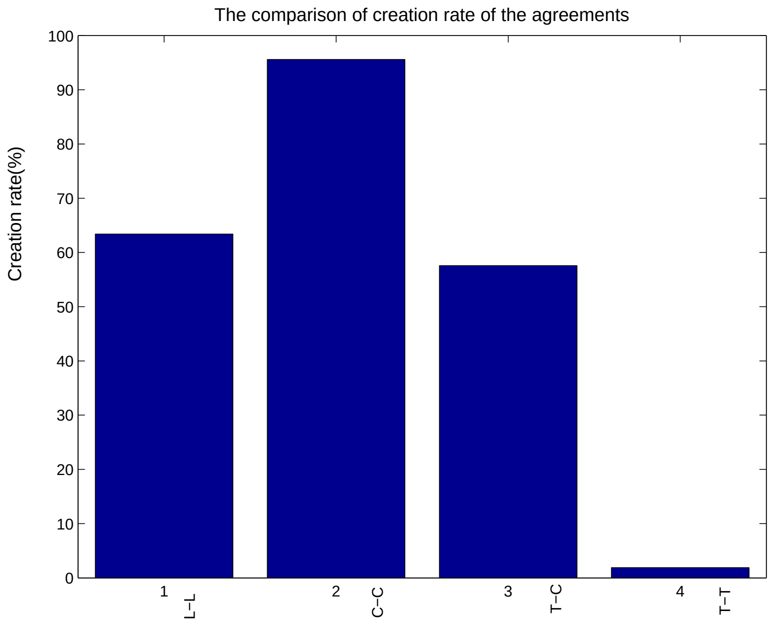

Figure 3.4: The Comparison of Creation Rate in Different Cases

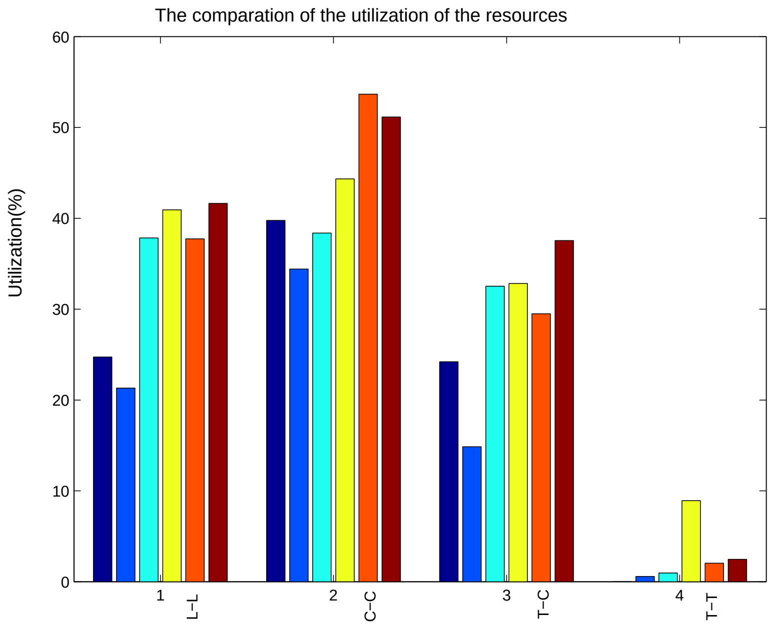

Figure 3.5: The Comparison of Utilization Rate in Different Cases

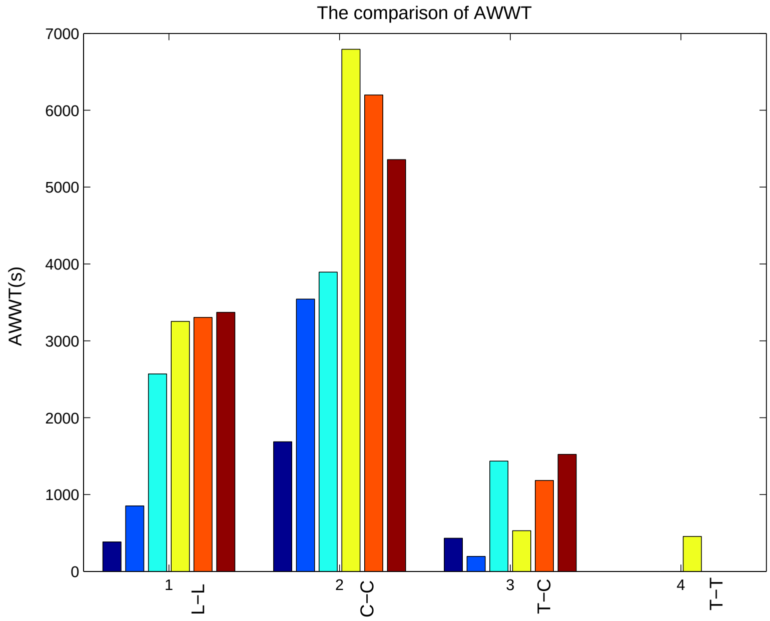

Figure 3.6: The Comparison of AWWT in Different Cases

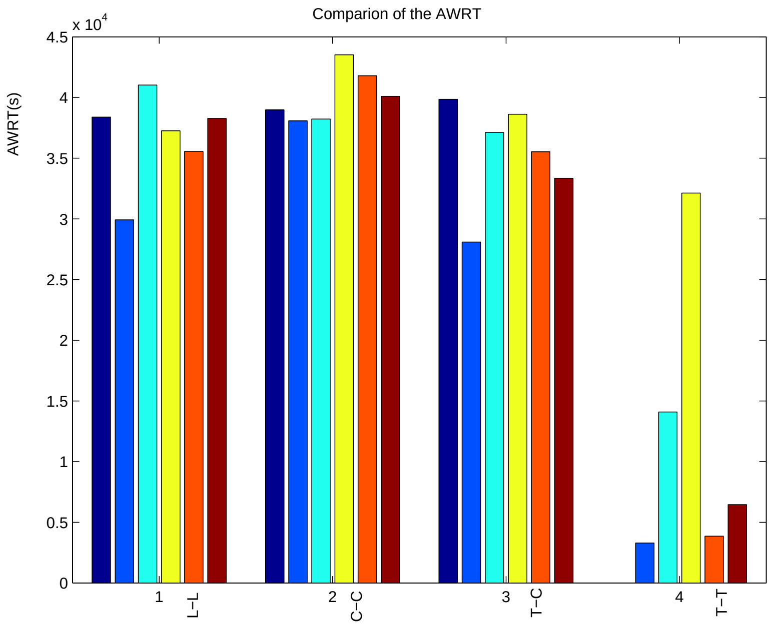

Figure 3.7: The Comparison of AWRT in Different Cases

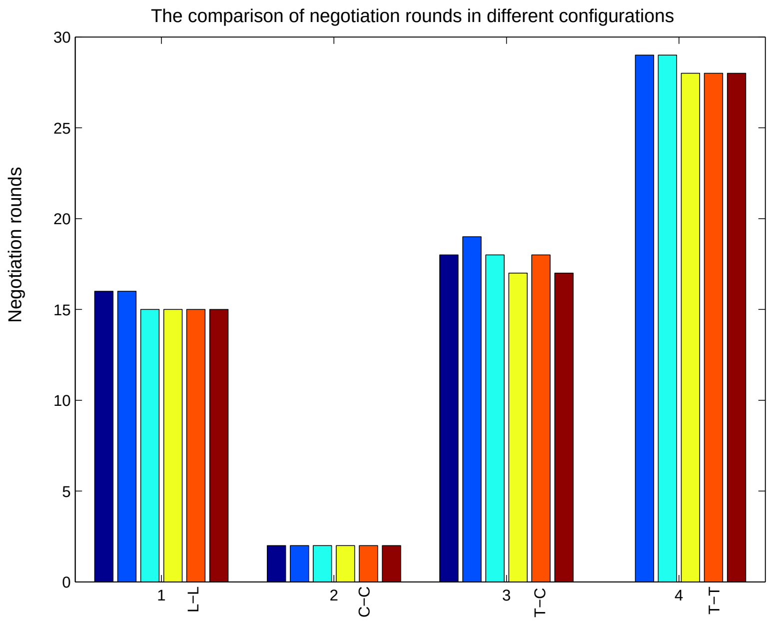

Figure 3.8: The Comparison of Negotiation Rounds in Different Cases

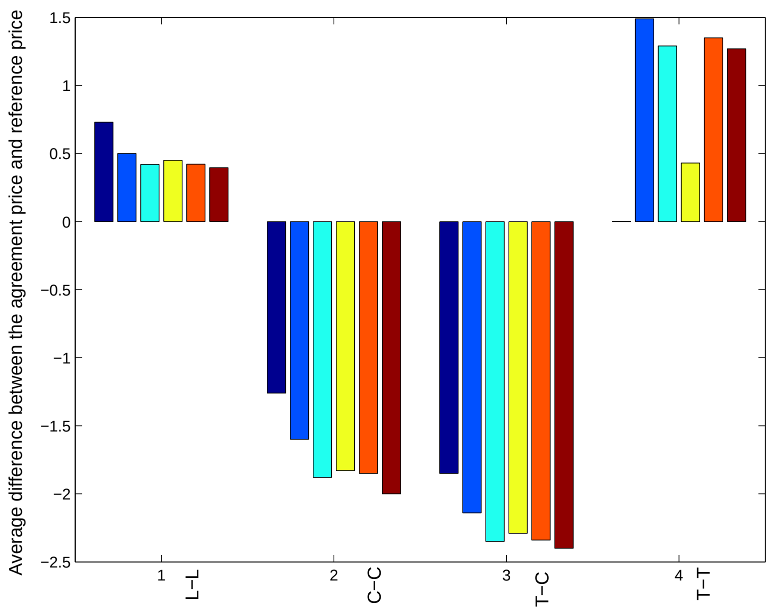

Figure 3.9: The Comparison of Average Difference Between AP and RP in Different Cases

However, the Figure 3.6 and Figure 3.7 show that in the C-C case, the AWWT and AWRT are high. This indicates that the conceding partners usually reach an agreement for this strategy, while the service quality for the user is relatively low as jobs are delayed. In all of these negotiation cases, the value of the AWWT and AWRT is comparable and in the same range as for Grid models which do not use negotiation models but conventional queuing systems, see e.g. [60]. That means, the presented model is feasible for real Grid infrastructure as it does not lead to any drawbacks in the performance results. However, the negotiated waiting time of the jobs will be guaranteed by the resource providers which is the anticipated quality of service level and can be seen as a major asset of such an approach.

In Figure 3.9 and 3.8, we can see that in the T-T case, although there are a very few number of successfully created agreements, the job users still have to pay higher cost and incurred much communication cost for negotiation. As shown in the picture, the agreement can only be created at the end of the user's negotiation span. In the T-C scenario, the succeeding rate of the created agreement is not so high, but the job users get on average cheaper offers from the resource providers. We can also see from these figures that in the L-L case, negotiation results are in the middle compared to the other cases.

In the time dependent negotiation strategies, the negotiation span can also influence the result of negotiation strategies. For example, in the C-C case, if we change the time span for negotiation for the resource providers to 20 seconds, the number of the successfully created agreements is the same while the agreement prices are lower as the provider concedes faster to his reservation value. Similarly, the resulting price is higher if the user has less time for negotiation. More simulations have been conducted, however we present only these excerpts of the results which show the feasibility of the model. From these simulations, we see that if the negotiation party is insistent on using a single strategy for the whole negotiation, it may not necessarily provide a higher utility. In order to get the most out of the negotiation, the negotiation parties will have to change their strategies dynamically during the process which will be addressed in the following chapters.

The current research in Grid computing shows that there is a trend for future resource management systems to include automatic management features for quality of service and cost consideration. As we can see from our experiments, the user can obtain quality of service and reliable agreement for the Grid jobs by applying the presented negotiation strategies. In our scenario, the expected waiting time is guaranteed by the resource provider. The simulation results show that the model can be used in the Grid scheduling environment. The presented results can be seen as first steps in analyzing the features and requirements for automatic negotiation strategies. They indicate that the negotiation overhead in terms of exchanged messages is manageable for practical application. The obtained agreement results can also be considered to be sufficiently for real world scenarios. But the time based negotiation strategies are quite simple and not flexible in the dynamically changing Grid environment, in the next chapter the learning based negotiation strategies which are much more flexible are introduced.

Chapter 4

Learning Based Negotiation Strategies for Grid Scheduling

The negotiation strategies proposed in Chapter 3 are not flexible enough in the dynamically changing Grid environment. In this chapter, learning-based negotiation strategies are proposed and examined similar to the proposed negotiation model in Chapter 3. Simulations have been conducted to evaluate the presented system. The results demonstrate that the proposed negotiation model and the learning based negotiation strategies are suitable and effective for Grid environments.

4.1 Learning Based Negotiation Strategies

In this chapter, the bilateral negotiation model and concurrent bilateral negotiation models are the same as in Chapter 3. The time dependent negotiation strategies [75] can be used to create the offers in the negotiation process, but they are not flexible enough for a dynamically changing Grid environment. Therefore, the learning-based negotiation strategies are proposed which allow the agents to dynamically adapt their α values (thereby different conceding strategies) according to their specific preferences.

The following learning-based negotiation strategies apply reinforcement learning algorithms [104, 105]. The Q-learning [106] algorithm was chosen because it is an online algorithm that does not require a model of the environment and thus it is well suited to dynamic and unpredictable Grid environments.

In the negotiation process, each negotiation agent uses a Q-learning algorithm to select the suitable time dependent negotiation tactic introduced before. In general, the agent's objective is to learn a decision policy that is determined by a so called state-action value function. The classical model of Q-learning consists of:

- a finite set S of states s of the concerned environment ($s \in S$);
- a finite set A of actions a that can be performed ($a \in A$);
- a reward function R: $S \times A \longrightarrow r$.

The agent's goal is to learn a policy: $S \longrightarrow A$ that maximizes the expected sum of discounted rewards V :

$$V[\gamma r_0 + \gamma^2 r_1 + ... + \gamma^n r_n] = V[\sum_{i=0}^{n} \gamma^i r_i] \tag{4.1}$$

where $0 \leq \gamma < 1$ is the discounting factor, the negotiation time is from 0 to n. The Q-learning algorithm is based on the estimated values of the agent's state (s)-action (a) pairs, called $Q(s,a)$ values. Based on these values, the agent updates its $Q(s,a)$ values using the formula:

$$Q(s,a) \longleftarrow Q(s,a) + \alpha[r + \gamma \times max_{a'}Q(s',a') - Q(s,a)] \tag{4.2}$$

where α is the learning rate which determines the rate of change of the estimation and $max_{a'}Q(s',a')$ is the value of the action that maximizes the Q function at state s. In the current research work, we use a ϵ-greedy [105] function that selects the action with the highest $Q(s,a)$ value. Using this approach, the learning agent behaves greedily most of the time, but every once in a while, with a small probability ϵ, it selects an action at random, uniformly, independently of the action-value estimates.

In order to use the Q-Learning algorithm, we have to identify the possible negotiation states and actions. For the *job users*, the states will be identified according to the number of currently available resources, the current remaining negotiation time and the types (tough or conceder, etc.) of the negotiation opponent. For the *resource provider*, the negotiation states can be identified considering the remaining negotiation time and the types (tough or conceder, etc.) of negotiation opponents. In order to identify the negotiation states, we divide the negotiation time into beginning part and ending part evenly. We can identify types of the negotiation opponents when negotiation time t is in the range of $[3, T]$ where T is the negotiation deadline of the negotiation party. For the job users, if $(2 * O^j_{t+1} - O^j_t - O^j_{t+2}) > 0$, then the resource provider uses conceder strategy; if it is less than or equal 0, the resource provider uses the tough or linear strategy, where O_t^j is the offer that job user received from the provider at time t. The number of available resources will be decided as high or low according to a specified threshold value.

As introduced before, we use the time dependent negotiation tactics to create the next offer. The actions for the negotiation parties are to select the proper parameters of α to produce the next offers.

Using the Q-Learning algorithms, the procedure of the adaptive negotiation algorithms is as follows:

- Initialize $Q(s, a)$ arbitrarily; specify the terminal states (i.e., agreement reached, deadline reached).
- Identify the current state s according to the parameters and get the reward signal r.
- Choose action a according to the ϵ-greedy policy, which means that it will choose the suitable α and generate the next offer using time dependent negotiation tactic.
- Terminate when terminal states are reached.

The reward functions of negotiation agents can be quite different. If both job users and resource providers want to create the agreement as soon as possible, the reward scheme for the job user is to reduce the weighted sum of the difference of the expected waiting time and the expected cost between the offers of the job user and the resource provider; the reward scheme for the resource provider is to reduce the difference of the cost offer between the offers of the job user and the resource provider. If they want to get higher utility and do not care whether they can create the agreements or not, they can use the opposite reward schemes. We just assume that the former is the *positive* reward scheme, while the later is the *negative* reward scheme. The different effects of these schemes will be evaluated using simulations. The rewards used in the current research (what the negotiation agents obtained during the negotiation processes) have different meanings from the expected discounted rewards mentioned before and they are used to guide the negotiation agents to choose the right actions at different negotiation times.

	Simulation Cases				
	1	2	3	4	5
Creationrate	85.92%	54.10%	42.40%	27.38%	42%

Table 4.1: The Rate of Created Agreements

4.2 Evaluation

Discrete event simulation has been used to evaluate the proposed negotiation model. In the following the simulation configuration is described and the simulation results are analyzed.

4.2.1 Simulation Results

We use the first 5000 jobs from the CTC workload traces [99] to do our simulation. We use the settings and configurations as defined earlier to compare the negotiation results in different simulation cases. The parameters used in Formula 4.2 are as follows: ϵ (we use the ϵ-greedy policy) is 0.2; learning rate α is 0.5; discount rate γ is 0.8; We assume that in the beginning of the negotiation, both of them use the tough behavior; if the negotiation agent does not use the learning algorithms, it will stick to the tough behavior. Here we only show some typical simulation cases.

The following simulation cases are considered. Case 1: Both of them use learning algorithms with positive reward scheme; Case 2: Both of them use learning algorithms with negative reward scheme and want to get higher utilities; Case 3: The job users do not use learning algorithm, while the resource providers adopt the learning algorithm with positive award scheme; Case 4: The resource providers do not use learning algorithms, while the job users use the learning algorithms with positive reward scheme. Case 5: The resource providers use the learning algorithms with positive reward scheme, while the job users adopt the learning algorithm and use the opposite reward scheme.

The success rates of negotiations in these 5 cases are shown in Table 4.1, from which we can see that when both of the negotiation parties want to create the agreement as soon as possible, the rate of successfully creating the agreements is the highest. In Case 4, the resource providers want to stick to higher prices and do not use the learning algorithms, the rate of successfully creating the agreements are much lower.

The simulation results are also shown from Figure 4.1 to Figure 4.6. In these figures, R1 to R6 stands for the resource 1 to resource 6.

From Figure 4.1 and Figure 4.2, we can see that in Case 1, the utilization rates of different resources are the highest; the required number of negotiation rounds is lowest. Therefore, using the learning algorithms with positive reward scheme, the negotiation parties can create agreements with a smaller number of negotiation rounds. In Case 3 and Case 5, the users stick to the tough negotiation strategies or the users use the learning algorithms with negative award scheme, so we can see that the negotiation rounds in these two cases are relatively more than other simulation cases. In Case 4, the resource providers insist on their original tough negotiation tactic without using the learning algorithms, the utilization rates of the resources are lower.

From Figure 4.3 and Figure 4.4, we can see that in Case 3, the users stick to the original tough behavior and they can obtain service with lower price and get the highest utility. In Case 5, the users use the learning algorithms with negative award scheme and they can also get higher utility and pay lower price. In Case 4, the resource providers stick to the original behavior, so the job users have to pay a very high price and get the lowest utility. As the simulation results in Chapter 3 showed that if both of the resource providers and the job users stick to the tough behaviors, the creation rate is only 1.88%, therefore, it is very difficult for them to create the agreements. Compared to the pure time based negotiation tactic, the learning based negotiation strategies are quite flexible in the dynamically changing Grid scenario.

From Figure 4.5 and Figure 4.6, we can see that the AWWT and the AWRT is comparable and in the same range as for Grid models which do not use negotiation models but conventional queuing systems. That means, the presented model is feasible for real Grid infrastructure as it does not lead to any drawbacks in the performance results. However, the negotiated waiting time of the jobs will be guaranteed by the resource providers which is the anticipated quality of service level and can be seen as a major asset of such an approach.

From these simulations, we can see that learning-based negotiation strategies are quite flexible and can actually be used in the dynamically changing Grid infrastructure. The negotiation parties can use learning-based negotiation strategies with different reward schemes depending on different objectives and preferences of the negotiation parties. If a negotiation party wants to create an agreement as soon as possible, it

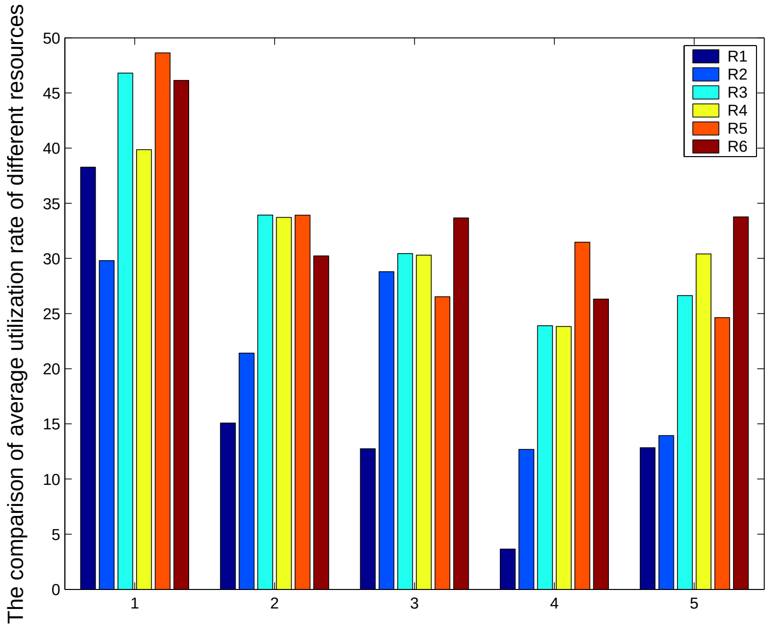

Figure 4.1: The Comparison of Average Utilization in Different Cases

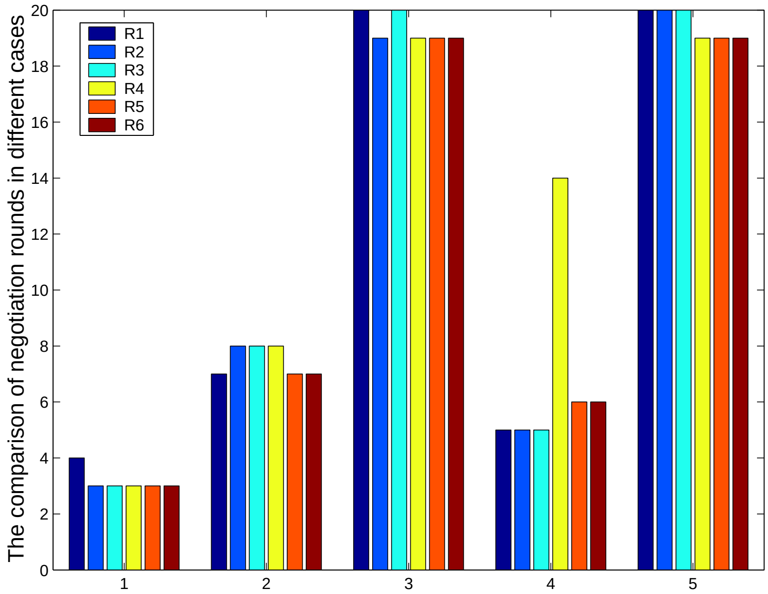

Figure 4.2: The Comparison of Average Rounds in Different Cases

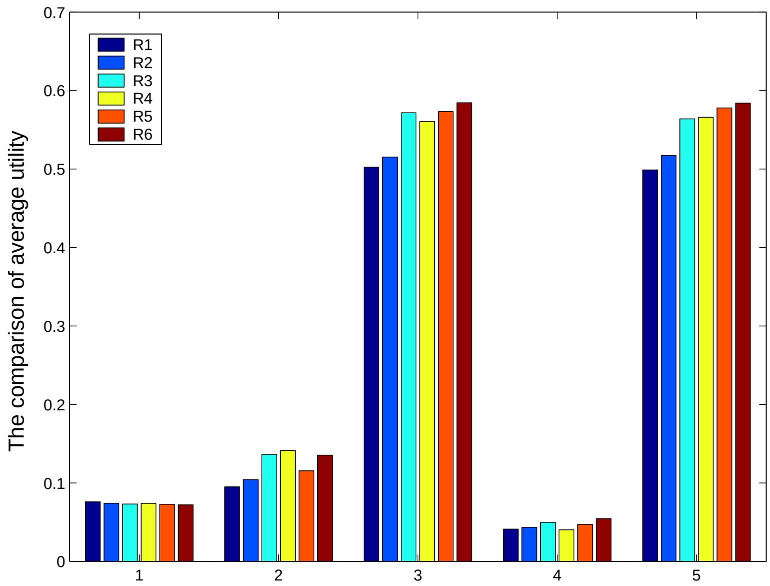

Figure 4.3: The comparison of Average Utility in Different Cases

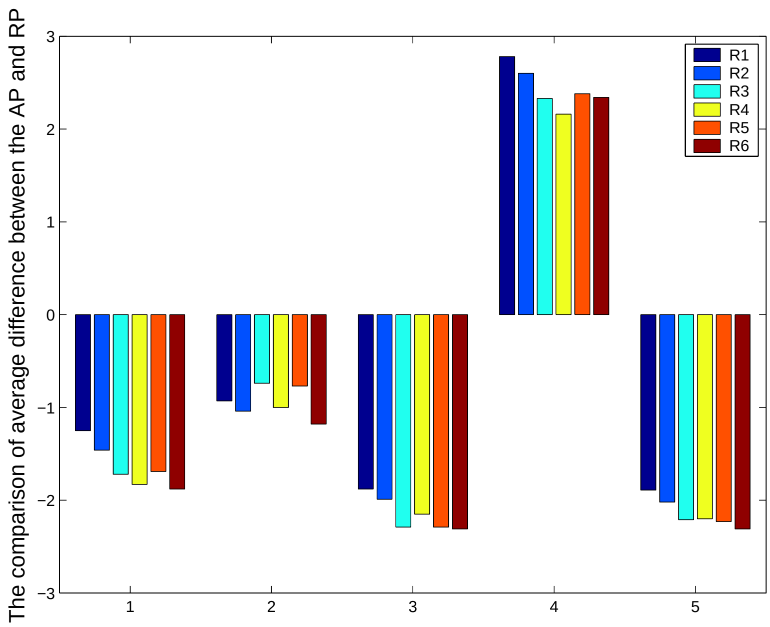

Figure 4.4: The Comparison of Average Price Difference in Different Cases

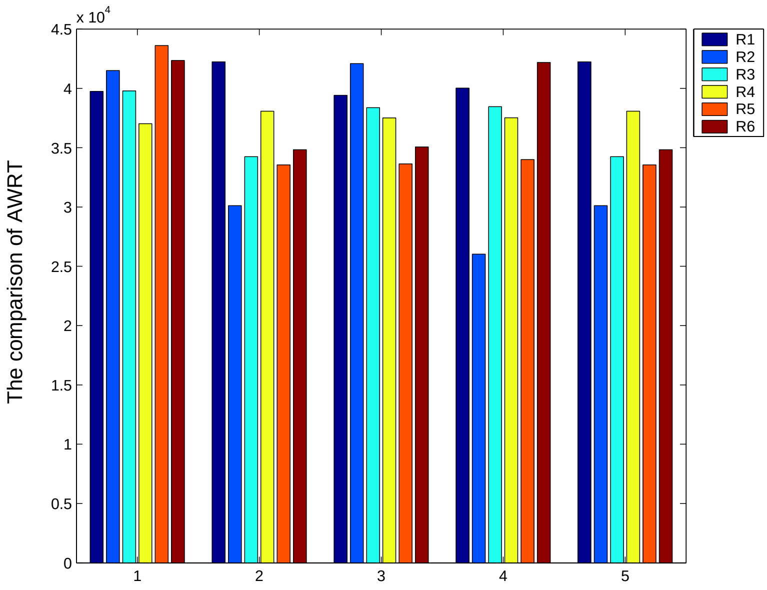

Figure 4.5: The Comparison of AWRT in Different Cases

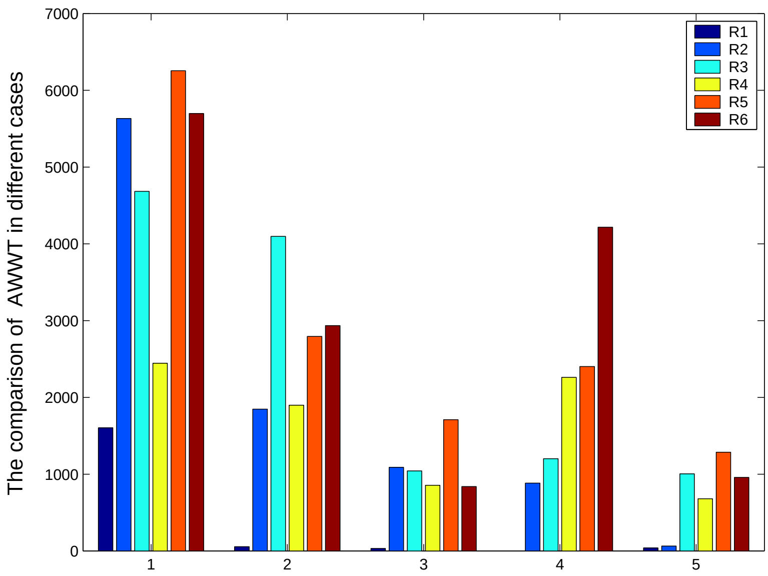

Figure 4.6: The Comparison of AWWT in Different Cases

can use a positive reward scheme; while a negotiation party wants to get higher utility and does not care if the process leads to an agreement creation, it can use a negative reward scheme. Of course, the resource provider can get higher price if he sticks to the tough behavior, but this will lose many chances of creating agreements, therefore it can not obtain higher utilization rate. The same applies the user, if he sticks to a tough behavior, he may not obtain services from resource providers even if it would yield a higher utility.

Q-Learning based negotiation strategies do not need the model of the whole system which are quite suitable for the Grid systems. Compared to Chapter 3, we can see that the learning based negotiation strategies are quite flexible and effective than the pure time based negotiation tactics. The results show that they can be applied in the practical use in automatic job scheduling. In the next chapter, we adopted the negotiation strategies using the so-called opportunistic functions.

Chapter 5

Negotiation Strategies Considering Opportunity Functions

In Chapter 4, the reinforcement-learning based negotiation strategies for the negotiation agents in Grid computing is proposed. In this chapter, the negotiation strategies considering opportunity functions are considered.

5.1 Introduction

In the reinforcement learning based negotiation strategies, the negotiation agent chooses the proper negotiation actions at different negotiation states according to the rewards got at each negotiation time. We identify the current negotiation states according to the number of the negotiation partners, the current negotiation time and the types of the negotiation opponents. In this chapter, we adopt and evaluate the negotiation strategies which consider opportunity functions of the negotiation agents as well as the difference between the offers and partners' offers.

As introduced before, a user will contact several resource providers to get the appropriate needed resources to execute its job in Grid computing, which is again the one to many multilateral negotiation type. The job agent has several options, that is, it has the freedom to choose which resource provider to negotiate with; but that does not indicate that the user will necessarily get the needed resource for sure if its private reservation value (e.g., maximum acceptable price or waiting time is too low) is not appropriate (there is no possible negotiation zone between two negotiation parties) or it always behaves very tough during the negotiation process. Therefore, in this chapter, we consider opportunity functions, that is, the agent will obtain a certain expected utility with at least one of its trading partners with a subjective probability.

5.2 Strategic Negotiation Model

In this chapter, the negotiation protocol and utility functions of the users are the same as in Chapter 3.

5.2.1 Negotiation Strategies Considering Time and Opportunity Functions

In the negotiation process, it is assumed that both of the negotiation agents behave according to *the good-faith bargaining principles* which means that it is usually not easily reversed [89]. Here, on the basis of the initial offer values, successive offers by sellers are monotonically decreasing while successive offers by the buyers are monotonically increasing. In order to create the agreement, both of the negotiation parties want to narrow the difference between the offers and counter offers with respect to different negotiation issues. In the strategic negotiation model, the negotiation agents can take different kinds of negotiation strategies developed in the agent community [21] to create the negotiation offer at different negotiation times. In the negotiation process, the strategies of the negotiation parties usually change dynamically based on the remaining available negotiation time. Of course, there is some other information which can be used to create the negotiation offers. Sim [107] proposed and analyzed the negotiation strategies which are for market-driven agents to make prudent compromises taking into account factors such as time preference, opportunity functions, competition factors.

The idea of opportunity functions is that in designing negotiation agents, an agent has the opportunity that it will obtain a certain expected utility with the subjective probability p with at least one of its trading partners in the concurrent negotiation threads. In the following, we briefly introduce the method of calculating p.

In a bilateral negotiation, the probability p of reaching consensus at an agent's own term can be derived as follows [107]:

Suppose agent B (the job agent) engages S_j (the resource provider) in round t. At any negotiation round t, $B's$ last proposal (bid) is represented by a utility vector $(V_t^{B\to S_j}, W_t^{B\to S_j})$ and $S_j's$ proposal (offer) is a utility vector $(V_t^{S_j\to B}, W_t^{S_j\to B})$. This means that: (1) if an agreement is reached in round t at $B's$ proposed bid, then B will get a payoff of $V_t^{B\to S_j}$ and S_j will get a payoff of $W_t^{B\to S_j}$ and (2) if an agreement is reached in round t at $S_j's$ proposed offer, then B will get a payoff of $W_t^{S_j\to B}$ and S_j will get a payoff of $V_t^{S_j\to B}$ (see Fig. 1 for illustration). From Fig. 1, it can be seen that $V_t^{B\to S_j} > W_t^{S_j\to B}$ for B and $V_t^{S_j\to B} > W_t^{B\to S_j}$ for S_j, i.e., B (respectively, S_j) will obtain a more favorable outcome if an agreement is reached at its proposed bid (respectively, offer). At t, if B accept $S_j's$ offer, it will obtain $W_t^{S_j\to B}$ with certainty. If B insists on its last proposal, and i) if S_j accepts it, B will obtain $V_t^{B\to S_j}$, and ii) if S_j does not accept it, it may be subjected to a *conflict utility* $U_B(D) = c^B = 0$, c^B is the worst possible utility for B , and $W_t^{S_j\to B} > c_B$. If B does not accept $S_j's$ last proposal, B may ultimately have to settle with lower utilities (the lowest possible being c_B). If there are changes in the market situation in subsequent cycles. For instance, B may face more competitions in subsequent cycles and may have to ultimately accept a utility that is lower than $W_t^{S_j\to B}$ (possibly as low as if the negotiation ends in disagreement). Not being able to acquire the resource it needs is the worst outcome for B (the job agent). If the subjective probability of B obtaining c_B is $P_{c,t}^{B\leftrightarrow S_j}$ (conflict probability) and the probability that achieves $V_t^{B\to S_j}$ is $1 - P_{c,t}^{B\leftrightarrow S_j}$, then based on Zeuthen's analysis [108], if B insists on holding its last proposal, it will obtain an expected payoff of

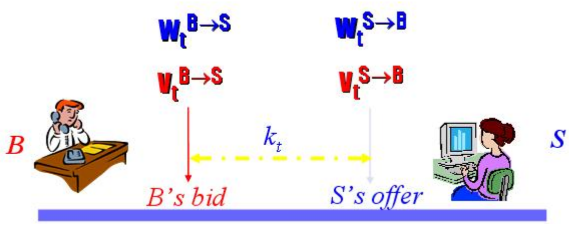

Figure 5.1: Tradeoff Between Bids and Offers for the Negotiation Parties

$$(1 - P_{c,t}^{B\leftrightarrow S_j})\times V_t^{B\to S_j} + P_{c,t}^{B\leftrightarrow S_j}\times c_B \tag{5.1}$$

Hence, B will find that it is advantageous to insist on its last proposal only if

$$(1 - P_{c,t}^{B\leftrightarrow S_j})\times V_t^{B\to S_j} + P_{c,t}^{B\leftrightarrow S_j}\times c_B \geq W_t^{S_j\to B} \tag{5.2}$$

Therefore,

$$P_{c,t}^{B\leftrightarrow S_j} \leq (V_t^{B\to S_j} - W_t^{S_j\to B})/(V_t^{B\to S_j} - c_B) \tag{5.3}$$

Consequently, the maximum value of $P_{c,t}^{B\leftrightarrow S_j}$ uses the highest probability of a conflict that may encounter at round t, given as follows:

$$P_{c,t}^{B\leftrightarrow S_j}{=}(V_t^{B\rightarrow S_j} - W_t^{S_j\rightarrow B})/(V_t^{B\rightarrow S_j} - c_B) \tag{5.4}$$

is a ratio of difference between two utilities. While $V_t^{B\rightarrow S_j} - W_t^{S_j\rightarrow B}$ measures the cost of accepting the trading agent's last offer (the spread k or difference between the (counter-)proposals of B and S_j), $V_t^{B\rightarrow S_j} - c_B$ measures the cost of provoking a conflict.

In a multilateral negotiation, if B has n_t^B trading partners, the aggregated conflict probability of B with all n_t^B partners is,

$$P_c = \prod_{j=1}^{n_t^B} \frac{V_t^{B\rightarrow S_j} - W_t^{S_j\rightarrow B}}{(V_t^{B\rightarrow S_j} - c_B)} \tag{5.5}$$

therefore, the probability that B will obtain a utility $V_t^{B\rightarrow S_j}$ with one of its negotiation trading partners is:

$$O(n_t^B, v_t^{B\rightarrow S_j}, \langle W_t^{S_j\rightarrow B}\rangle) = 1 - \prod_{j=1}^{n_t^B} \frac{V_t^{B\rightarrow S_j} - W_t^{S_j\rightarrow B}}{(V_t^{B\rightarrow S_j} - c_B)} \tag{5.6}$$

As explained before, the negotiation party will also modify the negotiation offer with the negotiation time going on. There are many ways of defining the function $\alpha_j^a(t)$ to model the effects of the remaining negotiation time. We also use the following function to calculate the $\alpha_j^a(t)$, see [109]:

$$\alpha_j^a(t) = k_j^a + (1 - k_j^a)(\frac{t}{t_{max}^a})^{1/\beta}, \tag{5.7}$$

where t_{max}^a is the deadline of the negotiation party a for the completion of the negotiation, t denotes the current time instant in the available negotiation time set, the parameter β is the degree of convexity that determines the type of the negotiation party in the time dependent strategy. Different β values yield different negotiation strategies. For the initial bargaining value k_j^a is used, for which the following relation holds $0 \leq k_j^a \leq 1$.

As pointed out in [107], there are several means of combining the time and the opportunistic function effects to create the offers for the negotiation parties, for instance, $0.5 * (T(t) + O(t))$, or $T(t) * O(t)$. Here we use the former one.

Assuming that P_c^t is the offered price at time t by the user, P_s^t is the offered price at time t by the resource provider; $T_c^t(job)$ is the proposed waiting time at time t by the user, $T_s^t(job)$ is the acceptable waiting time for the specific job at time t according to the current resource status considering the future reserved resource as well.

We assume that V_j is the utility function of the negotiation party which associates with the negotiation issue j and the $x_{a\to b}^t[t]$ is the offer provided by one party (denoted by a) to another negotiation party (denoted by b). The max_j^a is the maximum acceptable value for negotiation party a for negotiation issue j; a_j^t and b_j^t is the offers from negotiation party a and negotiation party b for negotiation issue j at time t respectively.

If V_j is decreasing:

$$x_{a\to b}^t[t] = a_j^t + 0.5((min(max_j^a, b_j^t) - a_j^t)) * (O(t) + \alpha_j^a(t)), \tag{5.8}$$

if V_j is increasing:

$$x_{a\to b}^t[t] = a_j^t + (1 - 0.5 * (\alpha_j^a(t) + O(t)))(min(max_j^a, b_j^t) - a_j^t), \tag{5.9}$$

Equations 5.8 and 5.9 represents the job user's strategy and the resource provider's strategy respectively.

As there are two negotiation issues involved in this negotiation process, we assume that if the offer in which one of the negotiation issues from the opponent is satisfied, then it will accept this value and will not further change it but only change the value of the remaining other issue in the following negotiation process. This is just a first heuristic to analyze the behavior of the negotiation strategies. In real life, negotiation issues will not be independent and thus the reaching of an acceptable offer is not easily achieved. For now, we accept that the negotiation issues are modified according to the previously made assumption on the monotonous increase/decrease by the parties.

5.2.2 Concurrent Bilateral Negotiation Model

The concurrent negotiation model is the same as introduced in Chapter 3. Because these negotiation threads are executed concurrently, it is very difficult to predict whether the user might achieve a better offer from another negotiation thread if there is already a suitable offer that could be committed to an agreement. In our model, we assumed that once an agreement is available, it will be created and committed. Of course, in a real life scenario the job agent might actually exploit the available time to find several offers

	Simulation Cases					
	1	2	3	4	5	6
Creation Rate	94.65%,	94.03%,	62.87%,	53.49%,	1.95%,	43.40%

Table 5.1: The Rate of Created Agreements in Different Simulation Cases

and decide at the end on the best offer. In Chapter 6 [110], we will analyze the results of tradeoff between the "best" and the "first available" agreement. In this chapter, for simplicity, we restricted our examination to accepting the first available agreement. If one negotiation thread is successfully negotiated, all of the other negotiation threads will be terminated. The agreement can then be used by provisioning and execution service to actually start a job on the local resource management system.

5.3 Evaluation and Simulation Result

We used the first 10000 jobs from the CTC workload traces [99] to do our simulation. We use the settings and configurations as defined earlier in Chapter 3 to compare the negotiation results in different simulation cases. We also use the evaluation criteria introduced in Chapter 3 to compare the simulation results.

In order to evaluate the simulation results, we compared the simulation cases with the associated simulation cases we did in Chapter 3 [109]. The following simulation cases are considered. Case 1: Both of them use the conceding strategy [109]; Case 2: Both of them use the conceding strategy and opportunistic functions; Case 3: Both of them use the linear strategy [109]; Case 4: Both of them use linear strategy and opportunistic functions; Case 5: Both of them use the tough strategy [109]; Case 6: Both of them use tough strategy and opportunistic functions;

The success rate of negotiations in these 6 cases are as the following tables. We can see that using the time and opportunity together can yield much higher creation rate in Case 6 than Case 5; in other simulation cases the creation rate are comparable.

A selection of results were shown from Figure 5.2 to Figure 5.7. R1 to R6 stands for the resources from 1 to resource 6 respectively. From these simulation results, we can see that the negotiation agents using the time and opportunist functions to narrow the differences between the offers and counter offers can achieve higher utilities than using negotiation strategies in Chapter 3 [109]. In the simulation cases which use the

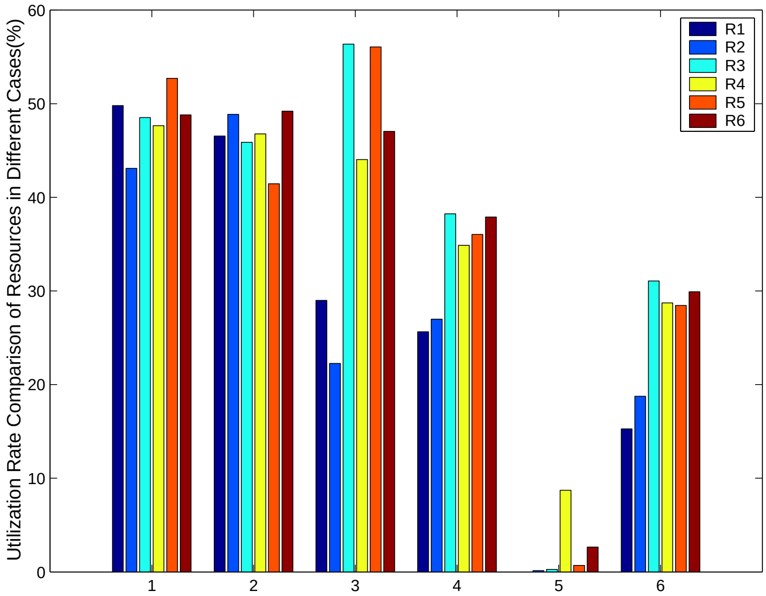

Figure 5.2: The Comparison of Utilization Rate in Different Cases

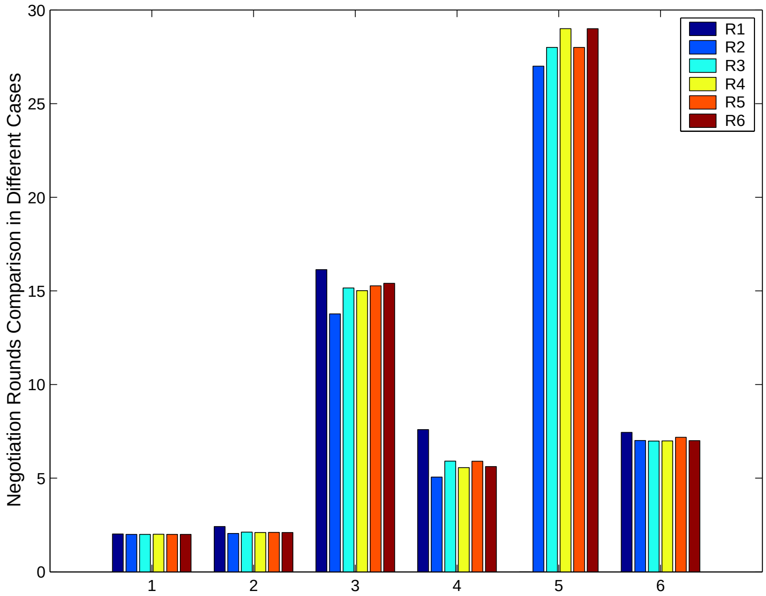

Figure 5.3: The Comparison of Negotiation Rounds in Different Cases

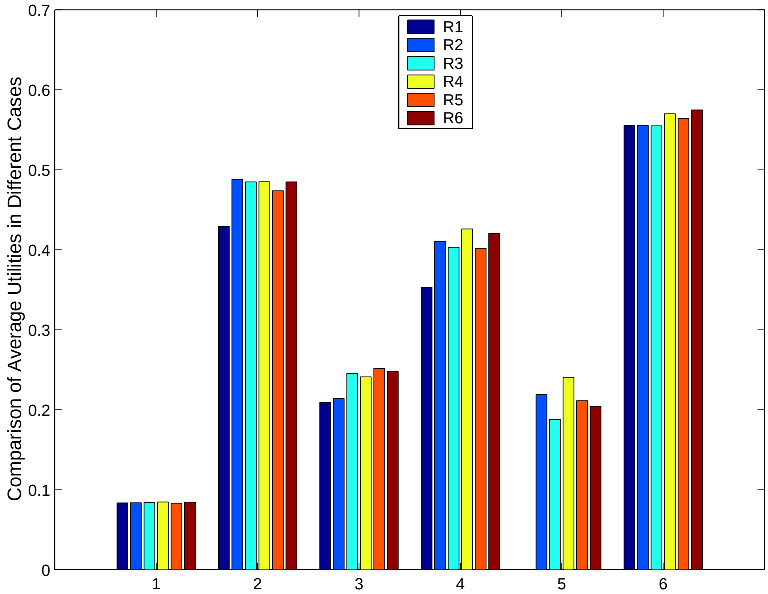

Figure 5.4: The Comparison of Average Utility in Different Cases

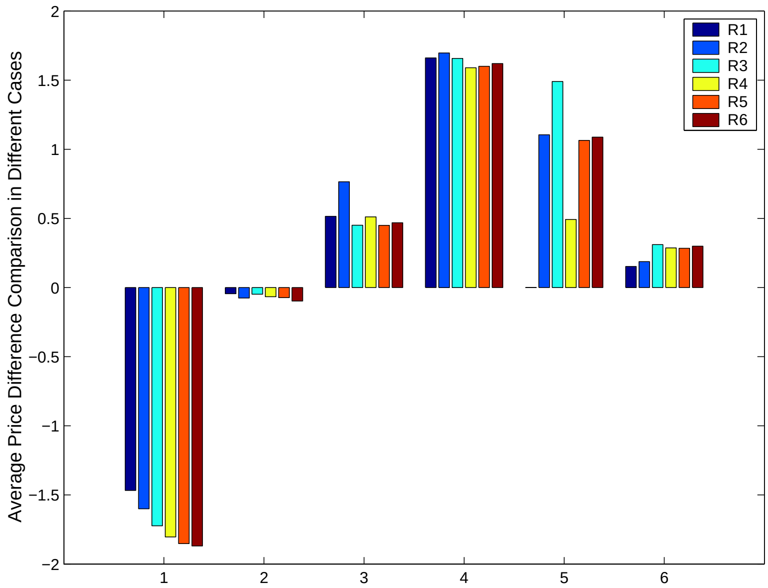

Figure 5.5: The Comparison of Average Price Difference in Different Cases

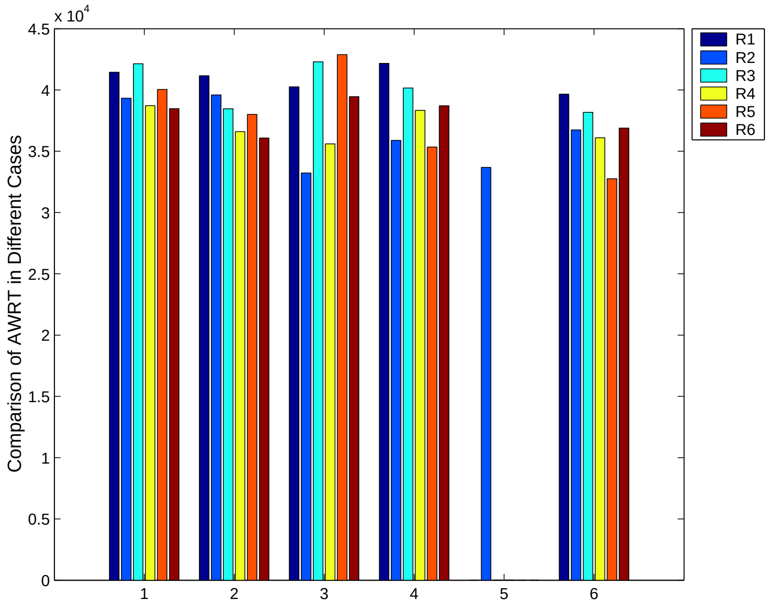

Figure 5.6: The Comparison of AWRT in Different Cases

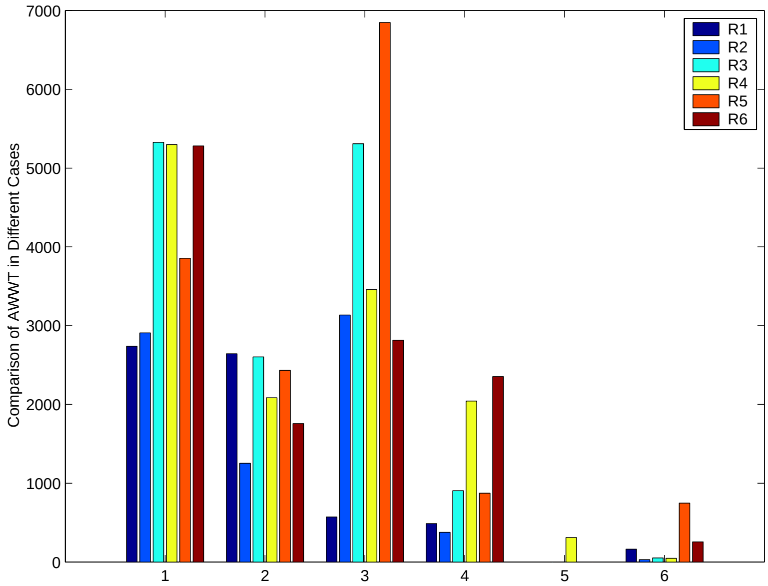

Figure 5.7: The Comparison of AWWT in Different Cases

time and opportunity functions, the AWWT is less than the cases in [109]. Except in the Case 5, there are no agreements created in R1, so the AWWT is 0, in other resources (except R2 and R4), the jobs can be started immediately due to the quite lower utilization rate. But the users usually pay more for the needed resources than in the simulation cases we did before as shown in the result figure 5.5. AWRT is comparable and in the same range as for Grid models which do not use negotiation models but conventional queuing systems. That means, the presented model can be considered feasible for real Grid infrastructure as it does not lead to any drawbacks in the performance results. To the contrary the negotiated waiting time of the jobs will be guaranteed by the resource providers which is the anticipated quality of service level and can be seen as a major asset of such an approach. Also we can see that the simulation cases using the opportunity functions can work with less negotiation rounds than using the pure negotiation tactic. An agent using opportunity function is more likely to reach a quicker agreement because it has higher chance of exploring more negotiation options. From these simulations, we can see that negotiation strategies considering time and opportunity functions are quite flexible and effective, and can actually be used in the dynamically changing Grid infrastructure.

Chapter 6

A Negotiation Model Supporting Co-Allocation and Trade-off for Grid Scheduling

As introduced before, the requirements of the users in a Grid environment are usually non-trivial. For example, several computers and network elements may be required in order to deal with the experimental data, while a large numerical simulation may require simultaneous access to multiple supercomputers and huge amount of storage as well as network resources. Therefore, support for the co-allocation between different resource providers is highly needed in Grid scenario. This chapter builds upon the results and concepts shown in Chapter 3 and 4. The original proposed model had several limitations. First, it does not easily support the co-allocation of resources from different providers. Another subject is the observation that in real life some users often do not know whether they can get "better" offers from other resource providers with the negotiation process going on. Thus, the provisioning of a preference function and the automatic selection of an offer might not be suitable. Therefore, we add also the optional concept of negotiating towards several possible negotiation offers and relaying the final decision to the user to select the "best" one. Therefore, we modify the state model of the agreement protocol by adding an additional non-binding state. This better supports the usage scenario of co-allocating resources, where it is essential that an "all or nothing" semantic is attained. Moreover, in the extended model the user party may not necessarily create and commit the "first" agreement but continue negotiation with other parties to find better solutions. To this end, we analyze the difference between "first" and "best" available offers in several simulation studies.

In this chapter, a negotiation model which supports the co-allocation between different resource providers in the Grid computing and the tradeoff for the user to choose between the "first" and "best" offer is proposed and evaluated using a discrete event based simulation.

6.1 Introduction

In order to fulfill the complex resource requirements of some users in Grid environments, support for co-allocation between different resource providers is needed. Here, it is quite difficult to coordinate these different services from different resource providers, because a Grid scheduler has to cope with different policies and objectives of the different resource providers and of the users. Agreement-based resource management is considered a feasible solution to solve many of these problems as it supports the reliable interaction between different providers and users. Here, a strategic negotiation model is needed to create such agreements between several Grid parties.

However, the co-allocation between different resource providers simultaneously or with time sequential dependence can not be easily supported by the model in Chapter 3, in which once there is chance to create the agreement, the agreement will be created and committed. There is no co-ordination between different resource providers. The negotiation party will create and commit the first available agreement without any tradeoff which can also be refined and extended. Moreover, it is considerably difficult to steer the negotiation towards agreements that fulfill a co-allocation request. A Grid scheduler has to cope with limited information and no insight on the negotiation policy of the provider agent.

6.2 Strategic Negotiation Model

In the current research work, we will consider the case of co-allocating several resources from different providers for concurrent timeslots. That is, all resources will have to be available at the same time. To this end, negotiations are conducted with several providers to reach independent agreements that facilitate this goal. Note, that we limit our examinations on this concurrent parallel co-allocation without loss of generality; the models can simply be extended to consider other time dependencies between allocations/agreements (e.g. in workflows).

The strategic negotiation model proposed in this chapter is the extension of the negotiation model in Chapter 3. In the same manner, a *bilateral negotiation model* is the building block of our *concurrent negotiations*, therefore we will briefly introduce this in the following.

6.2.1 Bilateral Negotiation Model

In order to support simultaneous co-allocation between different resource providers, common free intervals of the different resource providers should be found and committed simultaneously. Usually these common free time intervals can only be found after serval negotiation rounds as providers may not want to expose all information on their resource situation. So we adopted and modified Rubinstein's sequential alternating offer protocol in Grids, see [96]. There should be a non-binding state in which neither negotiation party needs to commit to an agreement unless one additional commitment signal is issued by one side. Another reason for introducing the non-binding state is that usually in the concurrent negotiation threads, the user expects to exploit several possible agreement offers. The user can select the current best offer which may not necessarily be "the first available offer" from the current possible offers. So we modify this protocol and add a pending state [34]. In this state both the resource provider and the resource consumer agree to make the agreement, but neither of them has to commit this contract. When one of the negotiation parties wants to commit this agreement, it has to notify the negotiation partner and also check whether the former created agreement terms are still valid or not. This allows that providers do not need to reserve any resources prior to this step. If the former terms are violated, then the agreement will not be committed if no re-negotiation process is conducted. After entering the commitment state, both of parties have to fulfill the agreement, otherwise the corresponding party may have to pay some kind of penalty due to the violation of agreement terms. The introduction of the non-binding state also makes it possible that the user can select the current "best" offer which may not necessarily be the "first" available offer from the current possible offers.

The utility functions of the users are the same as the models shown in Chapter 3. Also in this negotiation model, the negotiation parties do not know the opponents' private reservation information and their preferences/utility functions. The negotiation strategies used in Chapter 3 and Chapter 4 are also adopted in this chapter.

6.2.2 Concurrent Bilateral Negotiation Model for Co-Allocation

In the concurrent negotiation threads in which a single user is involved, the reservation value of the negotiation issues and preferences are the same. However, the user may adopt different strategies with respect to different negotiation partners. Furthermore, they might change their strategies during the negotiation process. Considering the difficulty and cost of co-allocation, if there is a single resource provider which alone satisfies the needs of the job user, our job agent will not consider further co-allocation. In order to coordinate these different and concurrent negotiation threads, there is a coordinator for every user that is in charge of every thread. There are several kinds of resource requests for the jobs that require co-allocation [61].

- In an **ordered request** the user will specify the resources from which the processors or other resources must be allocated. Ordered requests are used in practice when a user has enough information about the complete system to take full advantage of the characteristics of the different resources. For example, the data available at the different resources may dictate a specific way of splitting up an application.

- For an **unordered request**, the user only specifies the numbers of processors or the specific resources it needs in the separate resources, allowing the scheduler to choose the resources for the components.

- A **flexible request** specifies the total number of processors needed. The agent or the scheduler has the right to decide how to separate the whole requests into different parts and distribute them into specific suitable resources.

- For **total requests**, there is only one resource which executes the job. It will not need any co-allocation, which can be used to compare the different co-allocation schemes.

In some co-allocation processes, the data staging and replication, communication problem has to be considered. At the time of our research efforts, we only concentrate on the processor co-allocation. In our current research work, we considered the unordered request and the flexible request of the jobs. In the resource discovery and the pre-selection phase, information like the total number of nodes at a resource provider is considered available. That is, during the process only providers are queried which in general could provide an offer; however, the current availability due to already existing allocations is not known.

For the **unordered request** case, after the decision on required resources, the concurrent negotiation threads are started. The process to find a common time slot in different resources is non-trivial. The “all or nothing” atomic transaction is required that provides a solution for the whole job request; that is, it is not allowed that partial agreements are finally committed which do not jointly fulfill the original request. We achieve this goal iteratively; the whole process is two-phase based. The first phase is the negotiation process, in which the co-ordinate will check and monitor all of the negotiation thread. If all of them have created the agreements, all of the negotiation thread will be in the pending state. However, no individual commitments are made. First, the coordinator will check whether the created agreements in combination are feasible or not, that is whether the common time slot (the latest possible time) can be found between these different resources. If the common time slot can be found, all of the corresponding agreements will be considered for commitment and the second phase, the actual commitment phase begins. If there is no suitable solution with common timeslots, all of the agreements will be aborted. The latest timeslot start time is used as the common start time to restart the negotiation again. This heuristic considers that further into the future nodes usually become available, as most jobs are optimized for an early execution time. The re-negotiation process is performed to find common time slot. These whole processes must end within the available negotiation time span.

In order to accommodate more application scenarios, we also investigate the case that the requirements of the jobs are **flexible requests**. In this case, the job agent will contact different resource providers firstly. The resource agent will create the offer which includes the possible number of the CPU nodes at different time which is equal or later than the proposed start time for the jobs. According to the currently available resource information, the job agent will aggregate these different resources together in order to satisfy the users requirements. Considering the complexity of the co-allocation, it is assumed that the job agent will always use least number of the involved resources. Then after that, the job agent will divide the total number according to the current resource situation although the resource information is quite dynamic. Afterwards, the following negotiation process will be the same as the request type of the jobs are unordered request.

Also the negotiation model proposed in this chapter makes it possible for the user to make the tradeoff between the “best” and the “first” agreement. Because these negotiation threads are executed concurrently in the dynamically changing Grid environment, it is very difficult to predict whether the user might achieve a better offer from another negotiation thread if there is already a suitable offer that could be committed to an agreement. In the negotiation process, the job agent can have their own choices, that is, some job agents may decide that once there is an agreement chance available, it will be created and committed; while some job agents may actually exploit several offers and decide the “best” offer in the currently available possible agreement chances within their negotiation spans.

6.3 Evaluation

In order to evaluate the proposed negotiation model, discrete event simulation has been used as before. In the following the simulation configuration is described and the simulation results are analyzed.

6.3.1 Simulation Configuration

At the beginning of the negotiation, the parties will always make the offers which are most favorable to themselves. So we assume initial values of 0 for the bargaining value k_j^a of all the negotiation parties, therefore, the user will bid from the lowest price and the minimum acceptable waiting time, while the resource provider will bid from the maximum price. For performance analysis we assume a negotiation interval of 1sec between every negotiation round. In the following, we describe the models of the users and the resource providers. In order to evaluate the proposed negotiation model, we will compare simulation results in different simulation cases. The following parameters used for the learning based negotiation algorithms are the same as Chapter 4.

6.3.1.1 User Model

In our simulation, we consider parallel batch jobs in an online scenario. If there is no single resource provider which can fulfil the requirements of the job user, co-allocation between different resource providers will be needed. Typically, users will behave quite individually in the negotiation process. For our simulation, we assume two different kinds of user objectives: time-optimization and cost-optimization. Below are the parameters of the user modeling which have been applied for the simulation.

As introduced before, there are several types of the job requests for the co-allocation problem. We assume that the requests of the job users have been fixed before the negotiation, that is, the job user will specify the needed numbers in every resource, but the job agent can freely choose the needed resource provider. In the multi-site co-allocation problems, communication costs should be considered. In our current research work, the emphasis is on the negotiation models and negotiation strategies, the communication costs are ignored.

As mentioned in Section 6.2.2, some job users will create and commit the agreement once there is the chance of creating the agreement; while some users will exploit several agreement chances if they have within their negotiation spans. These two cases will also be simulated and compared. Below are the parameters of the user modeling which have been applied for the simulation.

- Negotiation span is uniformly distributed in $[0, 30]$ seconds.
- Maximum price acceptable for the user is uniformly distributed among $[4.0, 9.0]$.
- Acceptable waiting time for the job start is uniformly distributed in $[0, 36000]$ seconds.
- For the tough negotiator, β value is uniformly distributed in $[0.02, 0.2]$.
- For the conceding negotiator, β value is uniformly distributed in $[20, 40]$.
- Weights of time and price for the time-optimization are 0.8 and 0.2, while the weights of the time and price for the cost-optimization are 0.2 and 0.8. That is, we never optimize solely for price or cost.

6.3.1.2 Resource Provider

For the local resource management system a FCFS scheduling strategy with backfilling [102] is adopted which is common for parallel computers. There is no preemption allowed, which means that once a job is started, it will run to completion. The resources are all homogeneous and only differ in the number of available CPU nodes at each site. Different resource providers have different policies and different negotiation strategies. It is assumed that users, or their agents, will contact resource providers, which could fulfill their hard constraints and requirements individually or together (first selection process). That is, if a single resource provider can fulfill the requirements of the job user, then the concurrent negotiation threads between these resource providers and the job user will be constructed in a round robin fashion. Otherwise, the job user and the resource providers whose maximum resource numbers added together can satisfy the needs of the job user will construct the concurrent negotiation threads. The simulated hardware configurations of resource providers are consistent with actual configurations of the systems from which the real traces are originated. In this chapter, we present results for traces from the Cornell Theory Center [99] which had in total 512 CPU nodes. In our simulation we assume a Grid scenario with 8 different machines (parallel computer or cluster with a given set of CPU nodes) and therefore 8 resource providers. However, to stay consistent with the available workload from the CTC traces, the total number of nodes for all simulated machines is again 512 nodes. The number of nodes on each machine and the negotiation parameters for each resource provider are given below.

- The number of CPU nodes is 64 for each machine.
- Their different maximum prices per CPU time are {`8.2,8.0,7.5,7.6,7.4,7.5,7.7, 7.9`}.
- Their different minimum prices per CPU time are {`2.4,2.3,2.0,1.95,1.90,1.80,1.90, 2.0`}.
- Negotiation deadlines of different resource providers are all 30 seconds, which means that usually the resource provider will not opt out of the negotiation once the negotiation thread is created.

- For the conceding negotiator, β value is {`32, 35, 34, 38, 40, 40, 37, 36`}.
- For the tough negotiator, β value is {`0.03, 0.05, 0.04, 0.10, 0.05, 0.06, 0.07, 0.08`}

6.3.2 Simulation Results

Again, we used the first 5000 jobs from the CTC workload traces [99] to do our simulation. We compare the simulation results using the evaluation criteria identified in Chapter 3. Also we compare the creation rate for jobs that need the co-allocation to see the influence of the different co-allocation scheme.

6.3.2.1 Unordered Request Simulation Result

As introduced before, the different structure and size of the job have great influence for the co-allocation result [111]. We compare 2 main cases for the unordered request simulation result. The first case (*Case1*) is that the job user wants to use a minimal number of resource providers. The job user will divide the total needed number into different resource providers with respect to the maximum resource number of providers. In the second case (*Case2*) the job user will divide the total needed number into different resource providers according to half of the maximum resource number of providers.

For every main case, the following *subcases* are considered:

Subcase1: both of them use the conceding strategies.

Subcase2: both of them use the tough strategies.

Subcase3: both of them use the linear algorithms.

Subcase4: both of them use the learning algorithms from [112] and want to reach the agreement as soon as possible.

Subcase5: The resource providers use the learning algorithms and want to create the agreement as soon as possible, while the job users adopt the learning algorithm and want to get higher utilities.

Table 6.1 shows the creation rate in different simulation cases, where $c1co$ and $c2co$ stand for the rate of successfully created agreements supporting co-allocation in the subcases of Case 1 and Case 2 respectively, while $c1total$ and $c2total$ stand for the total rate of successfully created agreements in the subcases of Case 1 and Case 2

	Subcases				
	1	2	3	4	5
c1co	11.86%	0	10.17%	13.56%	6.78%
c2co	58.47%	0	39.83%	55.93%	19.49%
c1total	93.72%	2.14%	63.52%	84.42%	49.24%
c2total	94.84%	2.14%	64.14%	85.18%	49.56%

Table 6.1: The Rate of Created Agreements

respectively. From this simulation result (Table 6.1), we can see that, in *Case1*, there are very few number of successful created agreements for the jobs which need the co-allocation. Note, that the absolute (low) value of this percentage is not of concern and only the comparison between the values is of importance. The reason for the small number of successful agreement lies in the given deadline for the job execution. Raising the statistical distribution for this feature will automatically lead to higher rate for successful allocations; close to 100% if the constraints are set accordingly. However, such a constraint setting would not impose a challenge for the negotiation strategy and does not give further insight. In the *Case2*, the rate of totally created agreements and the co-allocation agreement rate is higher than in *Case1*. The simulation results show that requests for co-allocation provide more flexibility to allocate to resources, thus it is much easier to create successful agreements here. Also we can see that it is much harder to find the possible resources if both parties, resource providers and users, behave according to the tough negotiation strategy.

From Figure 6.1 to Figure 6.6, the simulation results for the Case 1 are presented. From Figure 6.7 to Figure 6.12, the simulation results for the Case 2 are shown. In these figures, R1 to R8 stands for the individual resources/machines in our configuration.

Figure 6.3, Figure 6.4, Figure 6.9 and Figure 6.9 demonstrate that the AWWT and the AWRT is comparable and in the same range as for Grid models which do not use negotiation models but conventional queuing systems. That means, the presented model can be considered feasible for real Grid infrastructure as it does not lead to any drawbacks in the overall performance results. However, now as an advantage the negotiated waiting time of the jobs will be guaranteed by the resource providers. This constitutes the anticipated quality of service level and can be seen as a major asset of agreement-based resource allocations. This feature is essential for co-allocation; in

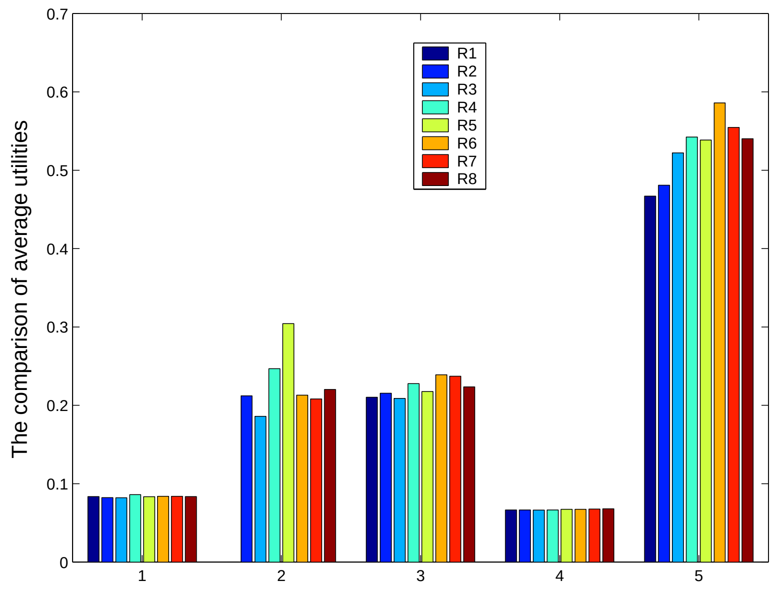

Figure 6.1: The Comparison of Utility in Different Subcases (Case 1)

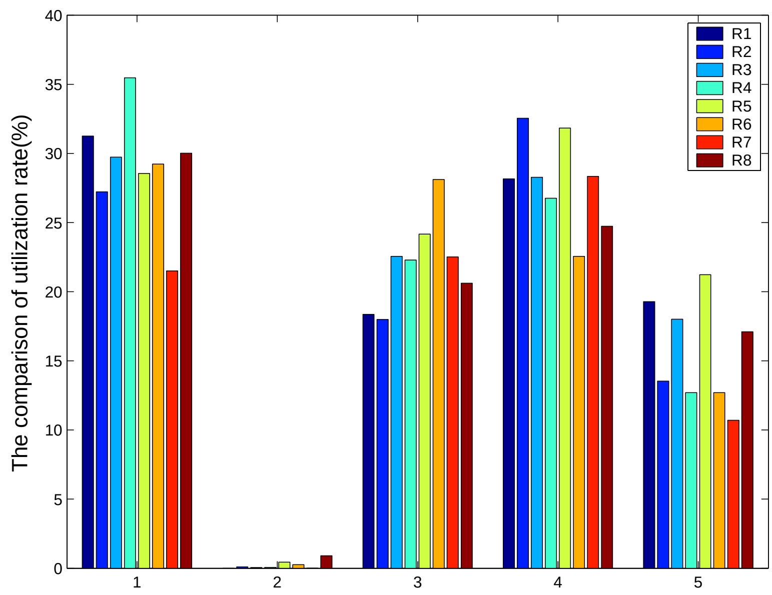

Figure 6.2: The Comparison of Utilization Rate in Different Subcases (Case 1)

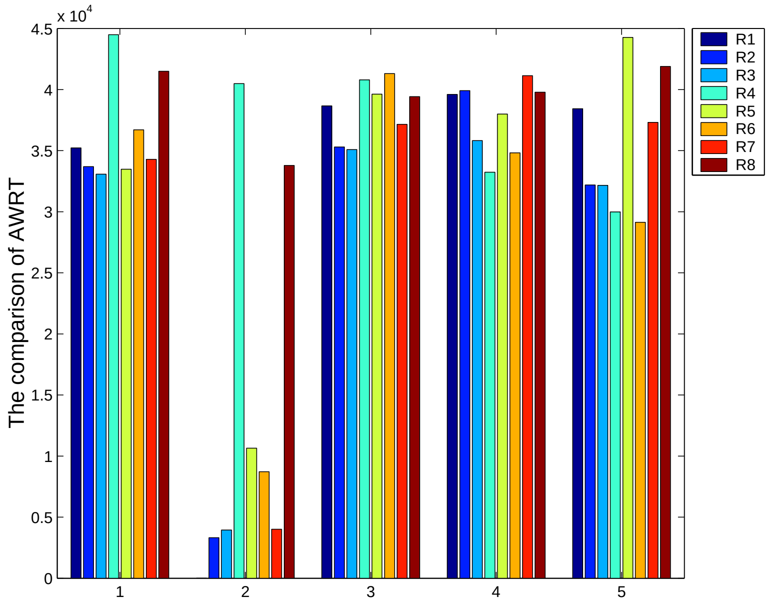

Figure 6.3: The Comparison of AWRT in Different Subcases (Case 1)

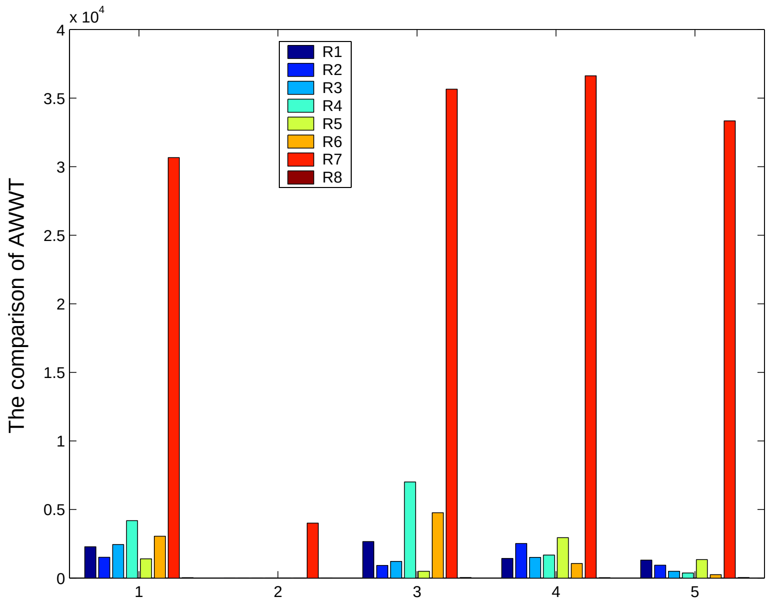

Figure 6.4: The Comparison of AWWT in Different Subcases (Case 1)

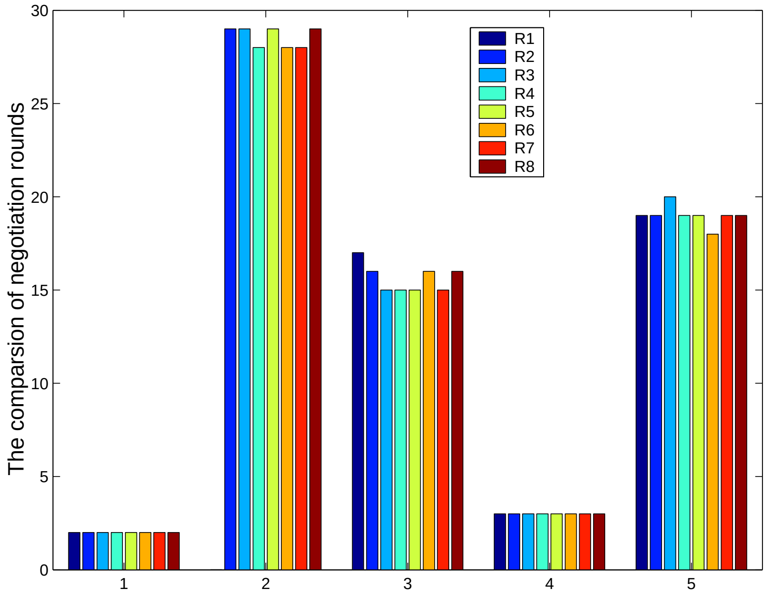

Figure 6.5: The Comparison of Average Negotiation Rounds in Different Subcases (Case 1)

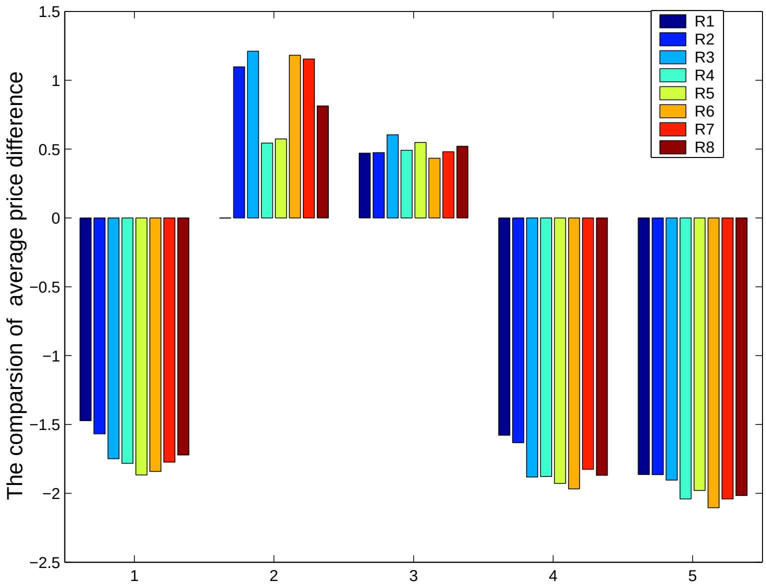

Figure 6.6: The Comparison of Average Price Difference in Different Subcases (Case 1)

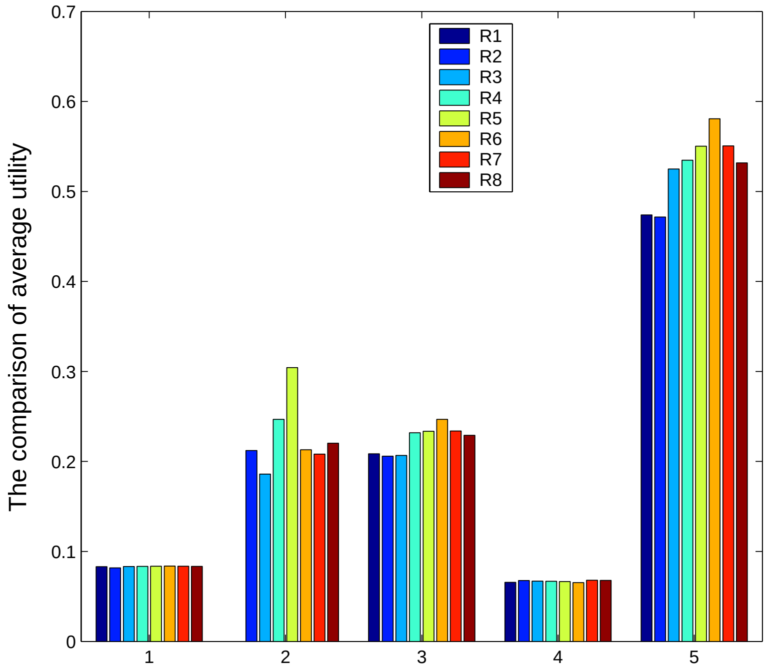

Figure 6.7: The Comparison of Utility in Different Subcases (Case 2)

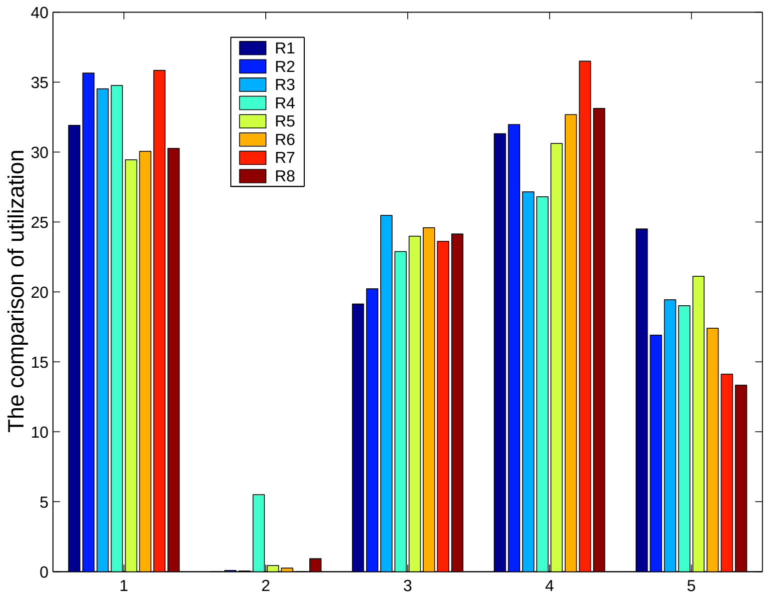

Figure 6.8: The Comparison of Utilization Rate in Different Subcases (Case 2)

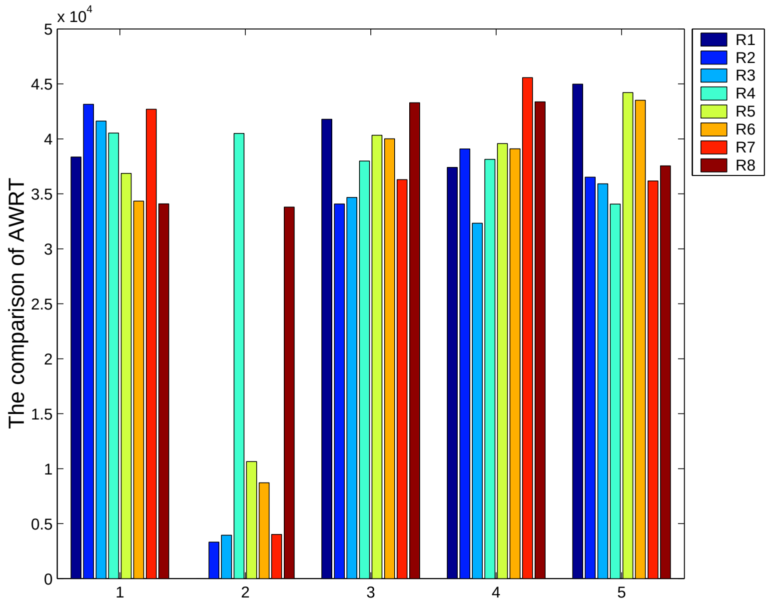

Figure 6.9: The Comparison of AWRT in Different Subcases (Case 2)

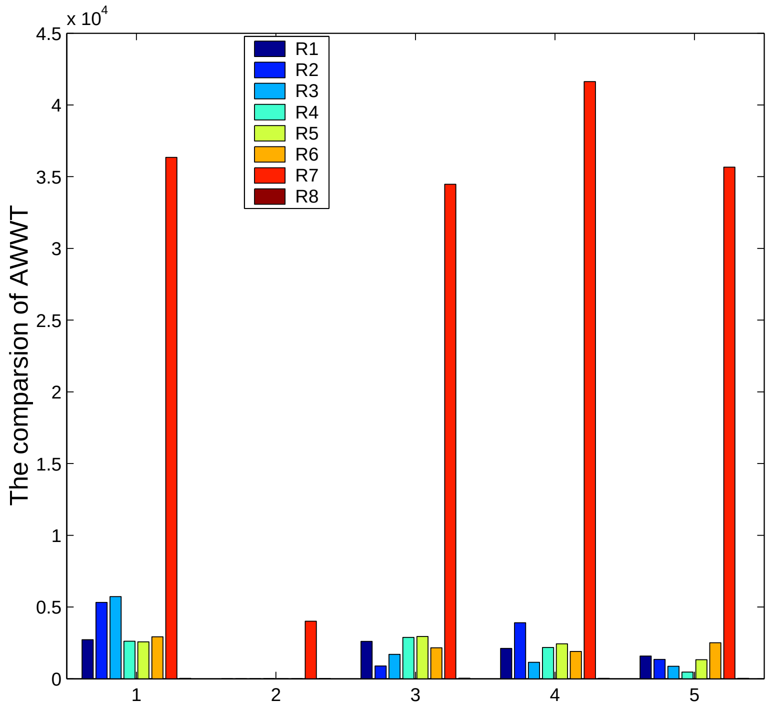

Figure 6.10: The Comparison of AWWT in Different Subcases (Case 2)

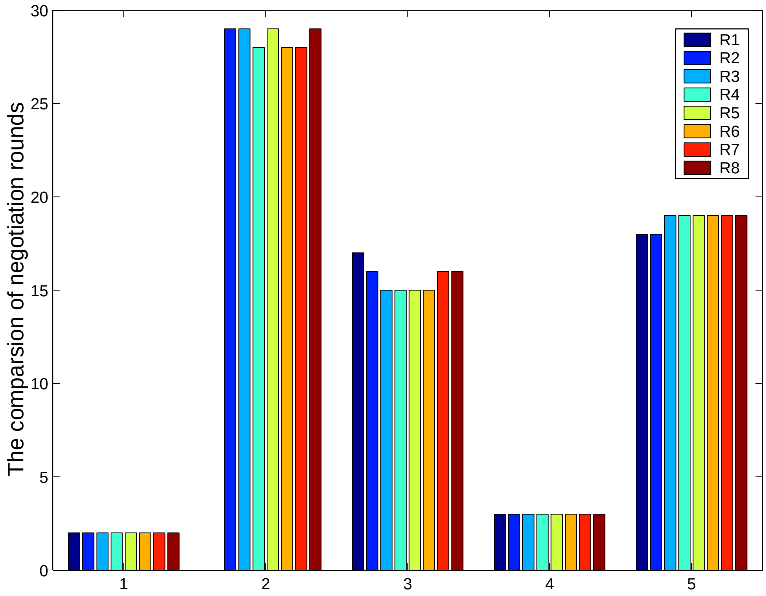

Figure 6.11: The Comparison of Average Negotiation Rounds in Different Subcases (Case 2)

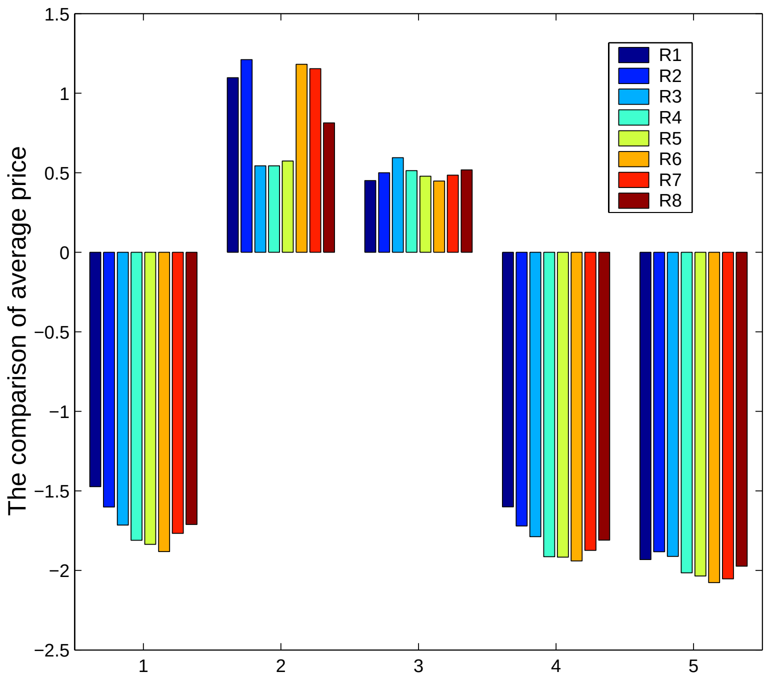

Figure 6.12: The Comparison of Average Price Difference in Different Subcases (Case 2)

order for the co-allocation to be successful, the resources which are involved in the co-allocation must be available at the same time for the specific job.

Figure 6.2, Figure 6.8, Figure 6.5 and Figure 6.11 show that in *Subcase1* of both Case 1 and Case 2, the number of successfully created agreements and the resource utilization is the highest; the required number of negotiation rounds is lowest; while in *Subcase2* of both Case 1 and Case 2, the number of successfully created agreements is quite low, and the required number of the negotiation rounds is highest. In *Subcase3* of both Case 1 and Case 2, the number of successfully created agreements and the required number of the negotiation rounds are in the middle of the *Subcase1* and *Subcase2*.

In *Subcase4* and *Subcase5* of both Case 1 and Case 2, the learning based negotiation strategies are used which are quite flexible and can change their negotiation tactics according to their own preferences in the dynamically changing Grid environment. Figure 6.1 and Figure 6.7 present that in *subcase5* of both Case 1 and Case 2, the user gets the highest average utility value.

The presented simulations are first steps to evaluate the usability of such negotiation strategies with co-allocation. These results indicate the strategies are indeed suitable for application. However, more simulations and practical analysis have to be conducted to evaluate the model. For example, in our presented simulation, additional overheads for co-allocation, e.g. due to communication and data staging, are not yet considered.

6.3.2.2 Flexible Request Simulation Result

In this subsection, we showed the simulation result for the flexible job request. In this kind of job request, the job agent will divide the total needed number to different resource providers according to the current resource situation at the requested time. The negotiation process has been introduced before. For the simulation, the following cases are considered:

case1: both of them use the conceding strategies.

case2: both of them use the linear algorithms.

case3: both of them use the learning algorithms from [112] and want to reach the agreement as soon as possible.

case4: The resource providers use the learning algorithms and want to create the agreement as soon as possible, while the job users adopt the learning algorithm and want to

	Simulation Cases			
	1	2	3	4
c3co	66.95%	40.68%	60.17%	21.19%
c3total	98.60%	66.64%	88.24%	66.64%

Table 6.2: The Rate of Created Agreements in Flexible Request Cases

get higher utilities.

Table 6.2 shows the rate of successfully created agreements in different simulation cases. *c3co* and *c3total* stand for the rate of successfully created agreements supporting co-allocation and the overall rate of successfully created agreements. The simulation results (Table 6.2 and Table 6.1) show that when the request of the job is more flexible, the creation rates of agreements are higher than the cases that the request of the job is not so flexible.

Figure 6.13 to Figure 6.18 show the simulation results in different cases. R1 to R8 stands for the individual resources/machines in our configuration. We can obtain similar results when they use different negotiation strategies, for instance, if both users and resource providers use the conceder strategies, then the creation rate of successfully agreements is highest.

6.3.2.3 Comparison Between "First" and "Best" Agreement

We also compare the case that user agents create and commit the "first" available (Chapter 4) and the possible "best" agreement offers. The following simulations are considered: Case 1: Both of them use learning algorithms with positive reward scheme, the job user will create and commit the first agreement; Case 2: Both of them use learning algorithms with positive reward scheme, the job user wants to get the best agreement offers; Case 3: The resource providers use the learning algorithms with positive reward scheme, while the job users adopt the learning algorithm and use the opposite reward scheme. The job user will create and commit the first agreement. Case 4: The resource providers use the learning algorithms with positive reward scheme, while the job users adopt the learning algorithm and use the opposite reward scheme and the job users want to exploit the "best" possible offers.

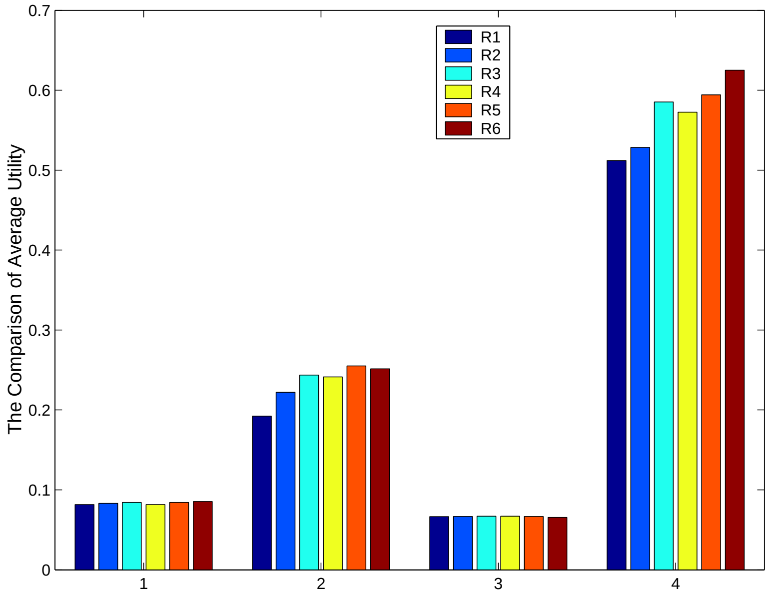

Figure 6.13: The Comparison of Utility in Different Subcases (Case 3)

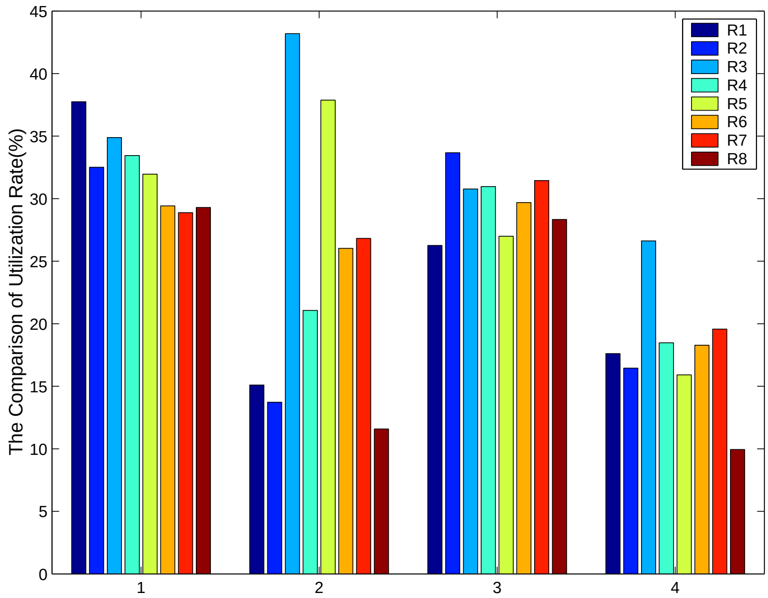

Figure 6.14: The Comparison of Utilization Rate in Different Subcases (Case 3)

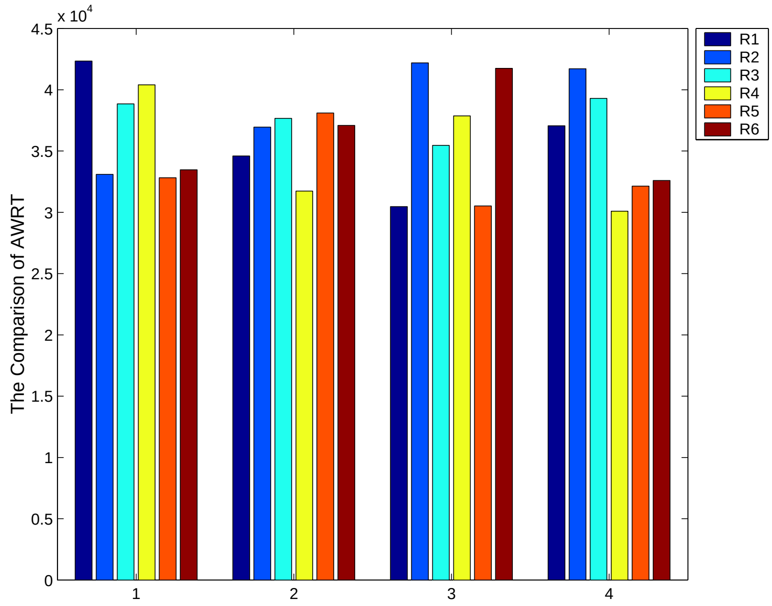

Figure 6.15: The Comparison of AWRT in Different Subcases (Case 3)

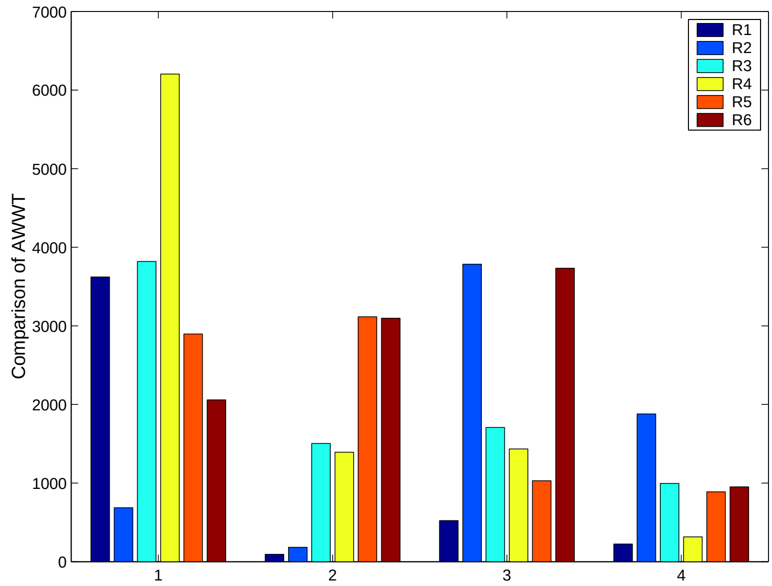

Figure 6.16: The Comparison of AWWT in Different Subcases (Case 3)

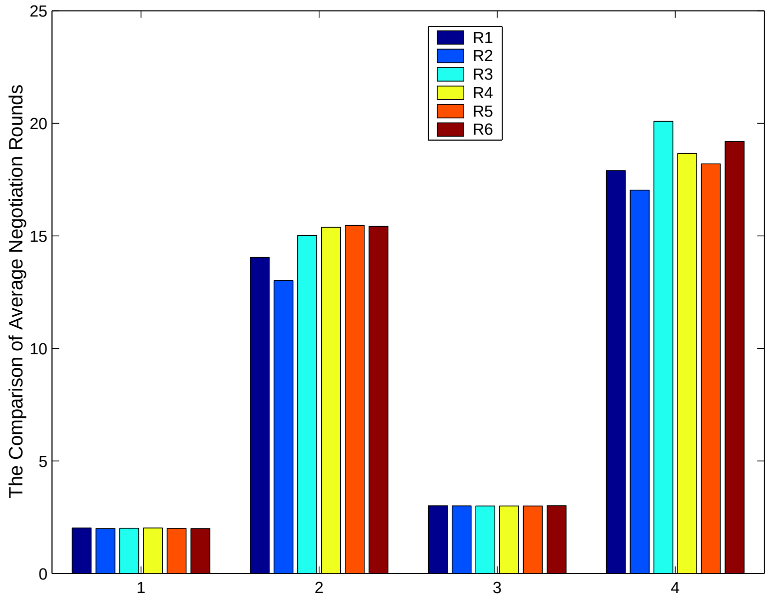

Figure 6.17: The Comparison of Average Negotiation Rounds in Different Subcases (Case 3)

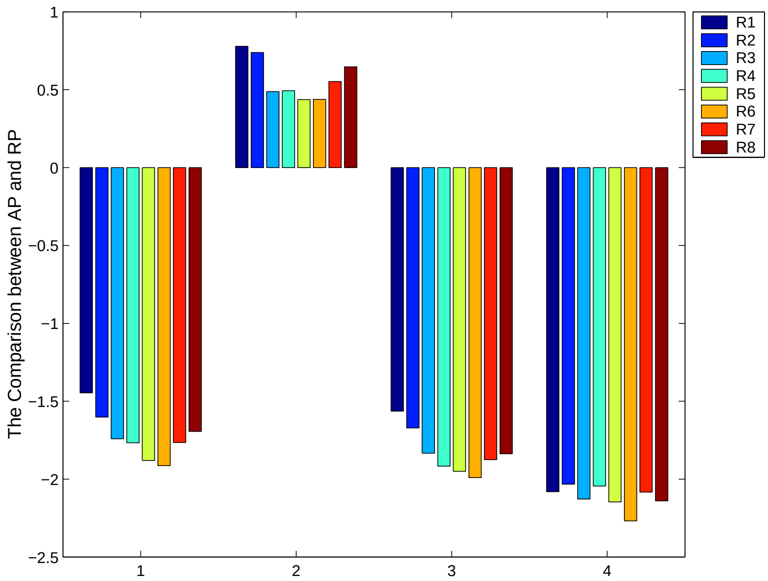

Figure 6.18: The Comparison of Average Price Difference in Different Subcases (Case 3)

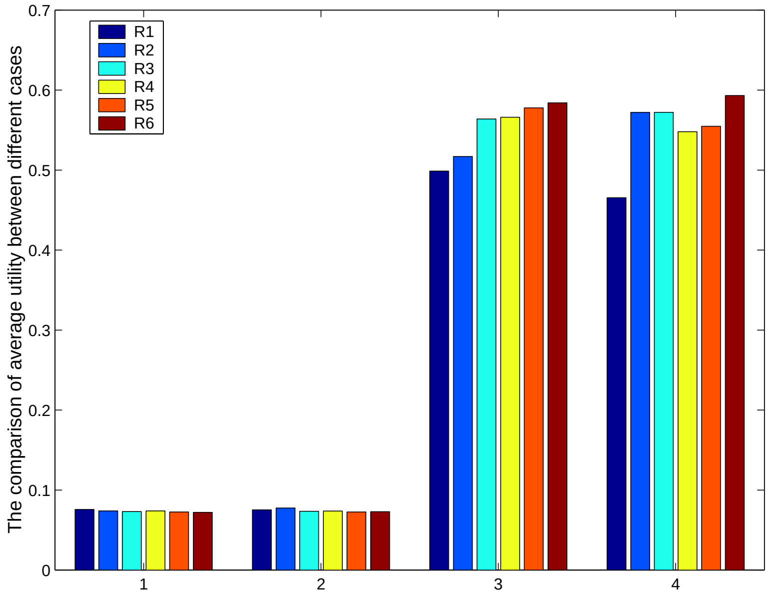

Figure 6.19: The Comparison of Utility in Different Cases

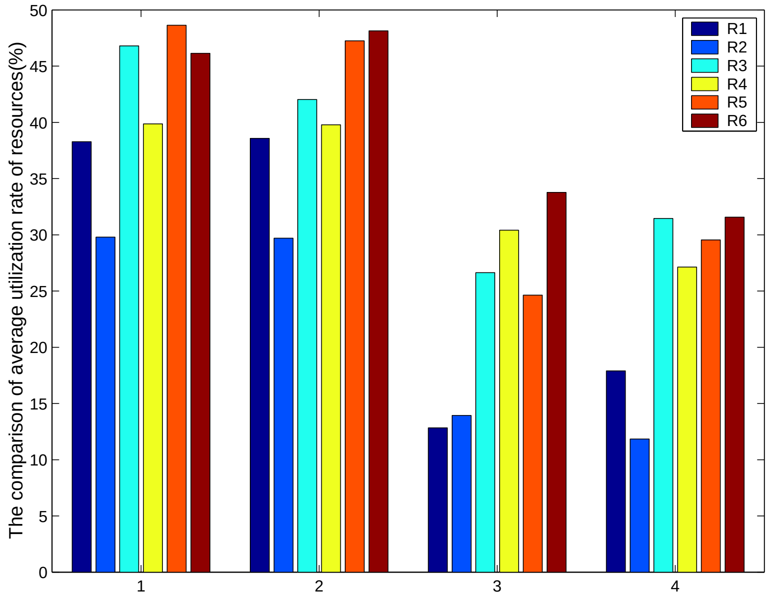

Figure 6.20: The Comparison of Utilization Rate in Different Cases

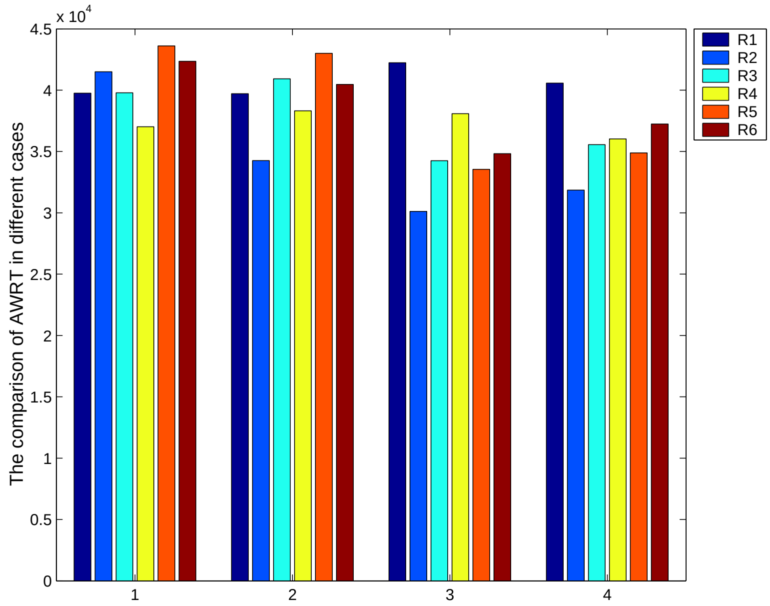

Figure 6.21: The Comparison of AWRT in Different Cases

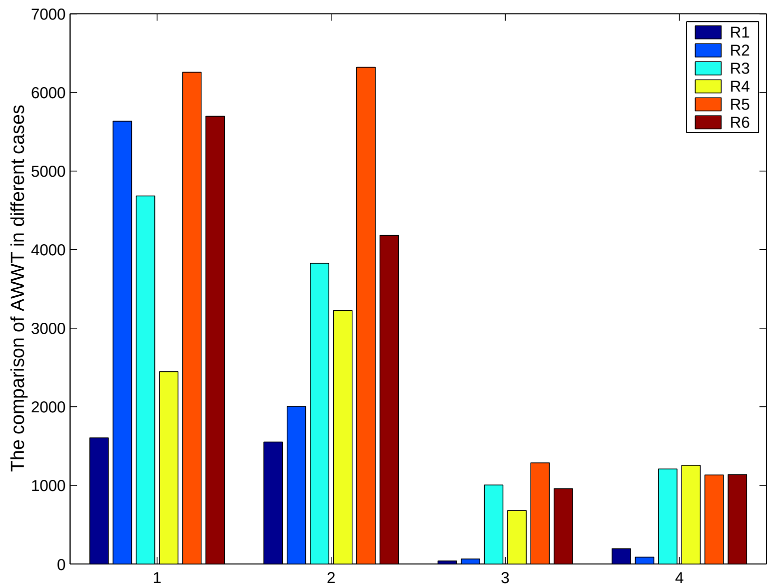

Figure 6.22: The Comparison of AWWT in Different Cases

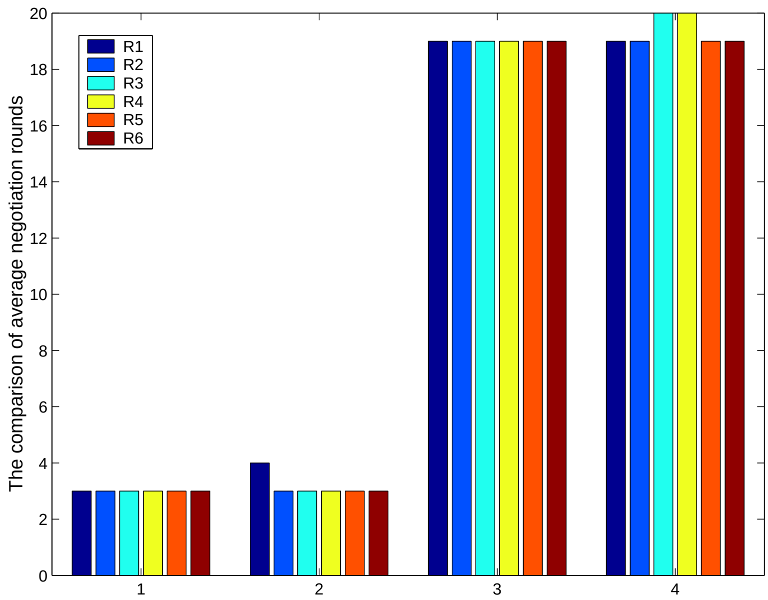

Figure 6.23: The Comparison of Average Negotiation Rounds in Different Cases

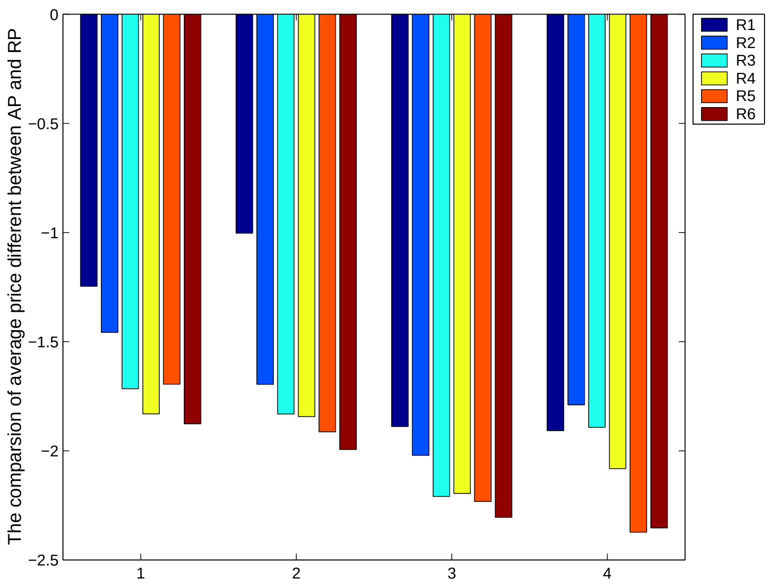

Figure 6.24: The Comparison of Average Price Difference in Different Cases

	Simulation Cases			
	1	2	3	4
Rate	85.92%	85.6%	42%	41.92%

Table 6.3: The Creation Rate of Agreements in Different Cases

	Result of Utility Comparison	
	Result 1	Result 2
Utility Greater	8	121
Utility Less	10	150
Utility Equal	4252	1815

Table 6.4: The Comparison of Utility Values

The simulation results are shown from Figure 6.19 to Figure 6.24. Table 6.3 presents the success rate of negotiations in these 4 cases. Between Case 1 and Case 2, Case 3 and Case 4, there are some common jobs which have been created the agreements and executed successfully although they may be run in different resources. In Table 6.4, we compared the number of jobs in Case 1 (Case 3) which have more utilities than Case 2 (Case 4). We call the comparison between Case 1 and Case 2 as the Result 1, and comparison between Case 3 and Case 4 as Result 2. From the simulation result, we can see that it is not always the case that the job users can get higher utilities even if they try to exploit "the best" offers within their negotiation spans in the dynamically changing Grid environment. And they may lose the chances of creating agreements.

In this chapter, a strategic negotiation model for Grids which extends the model proposed in Chapter 3 in order to support co-allocation between different resource providers is proposed and evaluated. First evaluation results, based on discrete event simulation, have been presented. From the simulation results, we can see that different kinds of job requests as well as their different negotiation strategies have great influences on the negotiation results. We also compared the tradeoff between the "first" and "best" agreements in this negotiation model.

Chapter 7

WS-Agreement Based Negotiation Protocol

As introduced in Chapter 2, in the Open Grid Forum (OGF), the working group on Grid Resource and Allocation Agreement Protocol (GRAAP) proposed the WS-Agreement protocol [34], which is the one-shot negotiation protocol and can only be used at the last stage in a transaction where the parties close their interactions with a contract specified as a WS-Agreement. One of the designing goals of the WS-Agreement protocol is that different negotiation models as well as negotiation protocols can compose the schemas defined by WS-Agreement. So the negotiation protocols as well as the negotiation frameworks based on WS-Agreement will be designed. The proposal of the WS-Agreement based negotiation protocol called WS-AgreementNegotiation [113] is the next step of the GRAAP. The draft of WS-AgreementNegotiation specification provides an additional layer to the WS-Agreement conceptual model, which defines some negotiation states, and port types, operations. This negotiation layer provides a Web service-based interface for negotiating an agreement so that it eventually satisfies both negotiating parties and becomes observed, and for renegotiating existing agreements after they have been observed.

As introduced before, considering the complex application scenarios in Grid computing, multi-rounded negotiation protocols which may be applied to different scenarios have to be proposed and investigated. In order to make the automatic negotiation possible, both of the negotiation parties should agree on the pre-defined negotiation models before the negotiation process begins. As introduced in Chapter 2, there are many negotiation models in the agent and economics communities which can be tailored to the different application scenarios in Grid computing. The research work in this thesis can be used as the alternatives for such negotiation models.

In this chapter, we can see that the negotiation protocols and the frameworks proposed in the current thesis work can be easily represented using the WS-Agreement semantics, therefore they can be easily adopted in the Grid computing practice.

7.1 Strategic Negotiation Framework

As shown before, there are three parts in the bilateral negotiation model that have to be considered [94]: 1) the negotiation protocol, 2) the used utility/preference functions for the negotiating parties, and 3) the negotiation strategy that is applied during the negotiation process.

For the negotiation protocol, we adopted and modified Rubinstein's sequential alternating offer protocol for Grids, see [96]. Every negotiation party will not know the preferences of the negotiation opponent.

7.1.1 Alternating Offers Messages

In the alternating offers protocol, there are several kinds of different messages: agreement template, offers, counter-offers, created agreement, etc. In order to use the WS-Agreement, the domain language should be defined. The Job submission description language (JSDL) [17] can be used as the domain language in the domain of job scheduling for Grid computing.

In the WS-Agreement context, the agreement provider will provide one agreement template. The agreement template is an XML document used by the agreement responder to advertise the types of offers it is willing to accept. Like an agreement document, the template is composed of a template name, a context element, and agreement terms, but additionally also includes information on agreement creation constraints to describe a range of agreements it might accept. For instance, the maximum number of CPU nodes, the capacity of the memory which can be provided by the resource provider. These agreement creation constraints must be observed by the negotiation opponent in the following negotiation process.

In the alternating offers negotiation protocol, the negotiation parties will do the negotiation in the specified negotiation time, so the negotiation time information should be identified in the agreement offers and counter offers, which can be included in the agreement context part of the WS-Agreement document. Negotiation issues, e.g., the waiting time and the cost to process the jobs can be expressed using the service properties terms. In the guarantee terms, the hard constraints, e.g., the needed CPU numbers must hold true in the following negotiation process. However, the specification of negotiability constraints in an offer does not state a promise that a replying offer fulfilling the constraints will be accepted. It is a voluntary disclosure of a preference to reduce the number of offers to be exchanged to create the agreements. In our current model, the private reservation negotiation information, e.g., the maximum waiting time of the job, the maximum acceptable price, is hidden from the negotiation opponent.

In the negotiation process, the users may have different objectives which can be described using the service level objectives. A service level objective [34] represents the quality of service aspect of the agreement. Syntactically, it is an assertion over the terms of the agreement as well as such qualities as date and time. In many cases, all service level objectives (SLO) will not carry the same level of importance. Relative "importance" terms can be used as a measure of importance with some specified values. So the different weights of different negotiation issues in our former models can be described using the "importance" value, but usually this information is private for the negotiation party.

As discussed before, the offers and counter-offers can be easily expressed using the WS-Agreement language and the domain specific languages as well the conditional language, etc. In the appendix, there are some examples of the offers and counter offers, created agreements. From this example, we can see that the specified hard constraints are observed in the created agreement.

7.1.2 Negotiation Protocol Semantics

As discussed before, the negotiation models should be agreed on by the bilateral negotiation parties before the negotiation process. However, WS-Agreement only defined the one-shot negotiation process. This is not sufficient to model a complete negotiation process; so additional port types and operations should be defined to support the alternating offers negotiation protocol.

There are several agreement states defined in the WS-Agreement proposal, e.g., pending, pendingandterminating, observed, observedandterminating, rejected, terminated, complete. But some following additional states should be added: offer, counter-offer, reject. In the offer state, the negotiation parties will propose the offer with the negotiation time going on, however, if the opponent reject it and the opponent will be in the counter-offer state and make an counter offer to it. Once the agreement is made, the negotiation will end. Also the port type of creating offer and counter offers can also be defined using the WS-Agreement semantics.

7.2 Decision Making Using Various Negotiation Strategies

Decision making process is private for every negotiation parties involved. In order to get the suitable offers in every negotiation rounds, the negotiation parties should have strategies or decision functions to create such offers in different negotiation time. At every negotiation time, the decision functions will analyze the counter offer of the negotiation opponent, calculate the current utility and decide the next action in the following negotiation time. The proposed negotiation strategies in this thesis have been evaluated using the traces.

In this chapter, we can see that the negotiation models proposed in this thesis can be easily expressed using the WS-Agreement semantics, therefore, they can be the alternatives for the WS-Agreement negotiation models.

Chapter 8

Conclusions and Future Directions

QoS and service level agreement support are important features for the next generation Grid. Resource management and scheduling is a very important component for the Grid infrastructure in which QoS and SLA should be provided. However, resource management in a decentralized infrastructure is a complex task as it has to cope with different policies and objectives of the different parties: providers and consumers/users. Agreement-based resource management is considered to solve many of these problems as the conflicts between the users and resource providers can be reconciled in a negotiation process. The whole scheduling process from the negotiation to the job execution in Grid computing should be automatically executed or with minimal human interaction, considering the potential scale of Grid systems and the amount of necessary transactions. Therefore, strategic negotiation models as well as the negotiation strategies must be proposed and evaluated to support the automatic negotiation and scheduling in Grid environments. In this thesis, the strategic negotiation models for the Grid scheduling are proposed and evaluated.

8.1 Summary and Conclusions

The contribution of this thesis has been the proposal of negotiation models as well as the negotiation strategies for Grid scheduling, which have been evaluated by discrete event based simulation approach using the workload traces and simulation results demonstrated their effectiveness in Grid environment. In this thesis, the following examinations and contributions have been made in detail:

- In Chapter 2, agreement based resource management approach has been analyzed. Requirements for the automatic negotiation problems in Grid computing have been introduced. Related works in the areas of economics and agent communities have been investigated.

- Several negotiation models and negotiation strategies have been proposed and examined. Simulation results demonstrate that these proposed negotiation models are suitable and effective for Grid environments.

- Firstly, in Chapter 3, strategic negotiation model using time-based negotiation strategies has been proposed and evaluated using discrete event based simulation technique.

- Then, the time-based negotiation strategies are quite simple and static which are quite limited in the dynamic changing Grid environment; so in Chapter 4, the learning based negotiation strategies have been investigated and evaluated, which are quite flexible and effective in the dynamic changing Grid environment. Also we have adopted the negotiation strategies considering opportunistic functions for Grid scheduling in Chapter 6.

- Thirdly, it is usually necessary that different resource providers co-allocate together to satisfy the complex requirements of the users, so in Chapter 5, strategic negotiation model supporting co-allocation and the tradeoff between "first" and "best" agreements in the Grid computing are also proposed and evaluated.

- Finally, the current work which contributes to the WS-Negotiation protocol has been analyzed in Chapter 7.

8.2 Future Work

In this thesis, the strategic negotiation models as well as the negotiation strategies can be applied in many application scenarios in Grid computing. However, much more work is still required to develop richer negotiation protocols and negotiation strategies that can be applied in more application scenarios in Grid computing. Some possible future directions are identified in this thesis and are presented as follows:

8.2.1 Leveled Commitment Contracts and Renegotiation

In the current research work, once the agreement is made, it will be observed and binding by either of the involved negotiation parties. In the dynamically changing Grid environment, there may be some future events, e.g., the user may find a better offer from other resource providers, which may have influences the decision of the negotiation parties. The **Contingency contract** has been suggested for use between self-interested agents when they have knowledge about the probabilistically known future events [89]. In these contracts, the payment will depend on future events, so the terms and payment as well as the penalty in case of breaching the agreement should be negotiated in the negotiation process. In the complex negotiation scenarios, these can make the situation more complex, but it may increase the utilities of the negotiation parties, so contracts not possible with full commitment protocols may become beneficial for both of the parties. But in the real-world negotiation parties usually can not know all possible future events and therefore can not use contingency contracts optimally. Even if the negotiation parties know all of the future events, it is very difficult for them to use this knowledge considering the huge computation efforts. Therefore, the contingency contract can only be used when the number of the future events are quite small, which usually can not be used with the increasing amount of the future events. There are also some other fundamental problems. For instance, the event may be observable by only one of the negotiation parties. Therefore, the leveled commitment protocol [86, 87] has been proposed which allows for the agent to decommit from a contract by paying a de-commitment penalty from a contract. The decommitting action can be done at any time. Instead of conditioning the contracts on future events, the leveled commitment protocol allows unilateral decommitting. If one party wants to decommit from the agreement, it simply pay the decommit penalty to the other negotiation party. In this case, the demcommit penalty is used to choose a level of commitment.

There are some advantages of introducing the level of commitment in the Grid scheduling and resource management considering the highly dynamic changing Grid environment (dynamic resource status, changing and complex requests from the user side, etc). For instance, the users may find better offers in the future; while the resource provider may be attracted by other users who are willing to pay more. In Chapter 5, we studied the results of the tradeoff between "best" and "first" agreement for the user, but we have not introduced the breaching of the formerly agreement terms and decommitment penalty. In the future, penalty strategies and polices should be supported and provided in the negotiation process in order to support the level commitment contracts and renegotiation.

8.2.2 Workflow Supporting

E-science and e-business are currently some of the main application scenarios in Grid computing. In these applications, workflows [114, 115, 116, 117] are typical use cases, therefore, workflow supporting is quite important for the Grid infrastructure. In a workflow, the time or other precedence between different procedures should be observed and guaranteed, therefore, in order to support the workflow execution, quality of service must be fulfilled. The strategic co-allocation negotiation models proposed in Chapter 5 can also be used to support the workflow execution, but such model should be evaluated using the typical workflow cases.

8.2.3 Multi-layer Negotiation

In Grid computing, the user can combine the agreements from different providers and make them into one whole agreement to support the job execution. Maybe the user agent can act as the broker for the resource provider, for instance, it can sell the using rights of the resources to other users. The negotiation models as well as the negotiation strategies should be provided to support such kind of application scenarios.

8.2.4 Policies Modeling and Research

Although the internal policies of the resource providers and users are not visible for negotiation partners, they have quite important influences for the agreement results. For the resource providers, the pricing models and the access rights with respect to different users should be constructed. For instance, the resource providers can predict the future demands of the users according to the demand history and the negotiation experience with the users. The policies of the resource providers and users have great influences on their negotiation strategies, therefore, the final agreement results and the performance of resource management and scheduling.

Appendix A

Appendix

A.0.4.1 Agreement Offer

```
<wsag:Name>agreementoffer</wsag:Name> <wsag:Contxt>
<wasag:negotiationtime> 1 </wasg:negotiationtime> <wsag:Context/>
<wsag:Terms> <wsag:All> <wsag:ServiceDescriptionTerm
wsag:Name="Host" > <job:Host> <job:hostname> Server </job:hostname>
</job:Host> </wsag:ServiceDescriptionTerm>
<wsag:ServiceDescriptionTerm wsag:Name="CPU">
 <job:CPU>
   <job:cputype>Intelx86</job:cputype>
   <job:speed>2</job:speed>
   <job:nodenumber>60</job:nodenumber>
 </job:CPU>
</wsag:ServiceDescriptionTerm> <wsag:ServiceDescriptionTerm
wsag:Name="memoryPerCPU" >
  <job:Memory>
   <job:number>200</job:number>
  </job:Memory>
</wsag:ServiceDescriptionTerm> <wsag:GuaranteeTerm
wsag:Name="BeginTime"> <wsag:Variables>
 <wsag:Variable wsag:Name="BeginTime" wsag:Metric="job:BeginTime">
   <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
 </wsag:Variable>
 </wsag:Variables>
 <wsag:ServiceLevelObjective>Waiting Time is equal 0
 </wsag:ServiceLevelObjective>
 <wsag:BusinessValueList>
  <wsag:Penalty>
   <wsag:AssessmentInterval>
     <wsag:Count>1</wsag:Count>
   </wsag:AssessmentInterval>
```

```
    <wsag:ValueExpression>5</wsag:ValueExpression>
  </wsag:Penalty>
  </wsag:BusinessValueList>
</wsag:GuaranteeTerm> <wsag:GuaranteeTerm wsag:Name="CPUNode">
<wsag:Variables>
  <wsag:Variable wsag:Name="Node" wsag:Metric="job:nodenumber">
    <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
  </wsag:Variable>
  </wsag:Variables>
  <wsag:ServiceLevelObjective>Node is equal 60
  </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm> <wsag:GuaranteeTerm
wsag:Name="CompuationCost"> <wsag:Variables>
  <wsag:Variable wsag:Name="cost" wsag:Metric="xs:float">
    <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
  </wsag:Variable>
  </wsag:Variables>
  <wsag:ServiceLevelObjective>cost is equal 2
   </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm> </wsag:All> </wsag:Terms>
```

A.0.4.2 Created Agreement

```
<wsag:Name>created agreement</wsag:Name> <wsag:contxt>
<wsag:negotiationtime> 10 </wsag:negotiationtime> <wsag:Context/>
<wsag:Terms> <wsag:All> <wsag:ServiceDescriptionTerm
wsag:Name="Host" > <job:Host> <job:hostname> Server </job:hostname>
</job:Host> </wsag:ServiceDescriptionTerm>
<wsag:ServiceDescriptionTerm wsag:Name="CPU">
  <job:CPU>
    <job:cputype>Intelx86</job:cputype>
    <job:speed>2</job:speed>
    <job:nodenumber>60</job:nodenumber>
  </job:CPU>
</wsag:ServiceDescriptionTerm> <wsag:ServiceDescriptionTerm
wsag:Name="memoryPerCPU" >
   <job:Memory>
    <job:number>200</job:number>
   </job:Memory>
</wsag:ServiceDescriptionTerm> <wsag:GuaranteeTerm
wsag:Name="BeginTime"> <wsag:Variables>
  <wsag:Variable wsag:Name="BeginTime" wsag:Metric="job:BeginTime">
    <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
```

```
 </wsag:Variable>
 </wsag:Variables>
 <wsag:ServiceLevelObjective>Waiting Time is equal 200
 </wsag:ServiceLevelObjective>
 <wsag:BusinessValueList>
  <wsag:Penalty>
   <wsag:AssessmentInterval>
     <wsag:Count>1</wsag:Count>
   </wsag:AssessmentInterval>
   <wsag:ValueExpression>5</wsag:ValueExpression>
 </wsag:Penalty>
 </wsag:BusinessValueList>
</wsag:GuaranteeTerm> <wsag:GuaranteeTerm wsag:Name="CPUNode">
<wsag:Variables>
 <wsag:Variable wsag:Name="Node" wsag:Metric="job:nodenumber">
   <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
 </wsag:Variable>
 </wsag:Variables>
 <wsag:ServiceLevelObjective>Node is equal 60
 </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm> <wsag:GuaranteeTerm
wsag:Name="CompuationCost"> <wsag:Variables>
 <wsag:Variable wsag:Name="cost" wsag:Metric="xs:float">
   <wsag:Location>/wsag:AgreementOffer/wsag:Terms/wsag:All</wsag:Location>
 </wsag:Variable>
 </wsag:Variables>
 <wsag:ServiceLevelObjective>cost is equal 5
 </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm> </wsag:All> </wsag:Terms>
```

References

[1] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *The International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200–222, fall 2001. 1

[2] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 2003. 1

[3] "The W3C Web Services Architecture working wroup public draft," http://www.w3.org/TR/ws-arch/, 2004.02. 1

[4] K. Czajkowski, I. Foster, and C. Kesselman, "Agreement-based resource management," *Proceedings of the IEEE*, vol. 93, no. 3, pp. 631–643, 03 2005. 1, 6, 8, 10, 20

[5] J. Nabrzyski, J. M. Schopf, and J. Weglarz, *Grid Resource Management: State of the Art and Future Trends*, Kluwer Academic Publishers, 2003. 1, 3

[6] "Platform Computing Cooperation LSF Administrator's Guide," Platform Computing Cooperation, 1996. 2

[7] "Open PBS," 11 2006, http://www.openpbs.org/. 2

[8] J. Balasayee, B. Blanchard, S. Kannan, and A. Tanishita, *Workload Management: SP and Other RS/6000 Servers*, IBM RedBook, 2000. 2

[9] "Sun Grid Engine," 10 2005, http://gridengine.sunsource.net/. 2

[10] D. G. Feitelson and L. Rudolph, "Parallel Job Scheduling: Issues and Approaches," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson

and L. Rudolph, Eds., pp. 1–18. Springer-Verlag, 1995, Lect. Notes Comput. Sci. vol. 949. 3

[11] D. G. Feitelson, L. Rudolph, U. Schwiegelshohn, K. C. Sevcik, and P. Wong, "Theory and Practice in Parallel Job Scheduling," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson and L. Rudolph, Eds., pp. 1–34. Springer Verlag, 1997, Lect. Notes Comput. Sci. vol. 1291. 3

[12] S. J. Chapin, W. Cirne, D. G. Feitelson, J. P. Jones, S. T. Leutenegger, U. Schwiegelshohn, W. Smith, and D. Talby, "Benchmarks and Standards for the Evaluation of Parallel Job Schedulers," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson and L. Rudolph, Eds., pp. 67–90. Springer-Verlag, 1999, Lect. Notes Comput. Sci. vol. 1659. 3

[13] I. Foster, N. R. Jennings, and C. Kesselman, "Brain Meets Brawn: Why Grid and Agents Need Each Other," in *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, Washington, DC, USA, 2004, pp. 8–15, IEEE Computer Society. 7, 11, 13

[14] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *The International Journal of Supercomputer Applications and High Performance Computing*, vol. 11, no. 2, pp. 115–128, 1997. 8

[15] "EGEE gLite Project," http://glite.web.cern.ch/glite/, 2006.11. 8

[16] "Condor Project," http://www.cs.wisc.edu/condor/, 2006.11. 9, 21

[17] "Job Submission Description Language WG (JSDL-WG)," http://www.ogf.org/gf/group_info/view.php?group=jsdl-wg, 2006.11. 9, 92

[18] "Globus Document," http://www.globus.org/toolkit/docs/development/4.2-drafts/, 2006.8. 9

[19] A. Cooke, A. J. G Gray, L. S. Ma, W. Nutt, J. Magowan, M. Oevers, P. Taylor, R. Byrom, L. Field, S. Hicks, J. Leake, M. Soni, A. Wilson, R. Cordenonsi, L. Cornwall, A. Djaoui, S. Fisher, N. Podhorszki, B. Coghlan, S. Kenny, and D. O'Callaghan, "R-GMA: An information integration system for grid monitoring.," *Lecture Notes in Computer Science 2888*, pp. 462–481, 2003. 9

[20] M. Wooldridge, "Agent-based software engineering," *IEE Proc Software Engineering*, vol. 144, no. 1, pp. 26–37, 1997. 12

[21] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge, "Automated Negotiation: Prospects, Methods and Challenges," *Int. Journal of Group Decision and Negotiation*, vol. 2, no. 10, pp. 199–215, 2001. 12, 18, 24, 57

[22] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995. 12

[23] T. W. Sandholm, "Distributed Rational Decision Making," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, G. Weiss, Ed., pp. 201–258. The MIT Press, Cambridge, MA, USA, 1999. 12, 13, 14, 15, 16

[24] N. R. Jennings, "On Agent-Based Software Engineering," *Artificial Intelligence Journal*, vol. 117, no. 2, pp. 217–296, January 2000. 13

[25] R. K. Dash, N. R. Jennings, and D. C. Parkes, "Computational Mechanism Design: A Call to Arms," *IEEE Intelligent Systems*, vol. 18, no. 6, pp. 40–47, 01 2003. 13, 14

[26] J.Nash, "Equilibrium points in n-person games," *Proc. of the National Academy of Sciences*, vol. 36, pp. 48–49, 1950. 15, 23, 27

[27] J.Nash, "Non-Cooperative Games," *The Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951. 15, 23, 27

[28] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *Journal of Finance*, vol. 16, pp. 8–37, 1961. 16

[29] D. Friedman and J. Rust, *The Double Auction Market: Institutions, Theories and Evidence.*, Addison-Wesley, 1993. 16, 23

[30] "Ebay," 12 2006, www.ebay.com. 16, 22

[31] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Transactions on Computers*, vol. C29, no. 12, pp. 1104 –1113, 1980. 16

[32] R. G. Smith, "The contract net protocol: high-level communication and control in a distributed problem solver," pp. 357–366, 1988. 16

[33] "Open Grid Forum," http://www.ogf.org, 2006.11. 17

[34] "Web Services Agreement Specification," 11 2006, http://www.ogf.org/gf/group_info/view.php?group=graap-wg. 17, 68, 91, 93

[35] "Web Services Addressing," ftp://www6.software.ibm.com/software/developer/library/wsadd200303.pdf. 17, 18

[36] "WSRF Specification Website," http://www.globus.org/wsrf, 2004.3. 17

[37] I. Foster, J. Frey, S. Graham, S. Tuecke, K. Czajkowski, D. Ferguson, F. Leymann, M. Nally, I. Sedukhin, D. Snelling, T. Storey, W. Vambenep, and S. Weerawarana, "Modeling Stateful Resources with Web Services v. 1.1.," March 2004. 17

[38] "WS-Resourceproperties," http://www.ibm.com/developerworks/library/ws-resource/ws-resourceproperties.pdf, 2004.3. 17

[39] "Web Services Base Faults," http://www-106.ibm.com/developerworks/library/ws-resource/ws-basefaults.pdf, 2004.3. 17

[40] "Web Services Service Group Specification," http://www-106.ibm.com/developerworks/library/ws-resource/ws-servicegroup.pdf, 2004.3. 17

[41] "Web Services Resource Lifetime," http://www.ibm.com/developerworks/library/ws-resource/ws-resourcelifetime.pdf, 2004.3. 17

[42] "WS-Base Notification," ftp://www6.software.ibm.com/software/developer/library /ws-notification/WS-BaseN.pdf, 2004.3. 18

[43] "Web Services Brokered Notification," ftp://www6.software.ibm.com/software/deve -loper/library/ws-notification/WS-BrokeredN.pdf, 2004.3. 18

[44] "Web Services Topics," ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-Topics.pdf, 2004.3. 18

[45] "Publish-Subscribe Notification for Web services," http://www-106.ibm.com/developerworks/library/ws-pubsub/WS-PubSub.pdf, 2004.3. 18

[46] "Grid Resource Allocation Agreement Working Group in OGF," http://www.ogf.org/gf/group_info/view.php?-group=graap-wg, 2006.11. 19

[47] "OGSA Basic Execution Services WG (OGSA-BES-WG)," https://forge.gridforum.org/sf/projects/ogsa-bes-wg, 2006.11. 19

[48] "Microeconomics," http://en.wikipedia.org/wiki/Market_form, 2006.08. 19

[49] R. Buyya, *Economic-based Distributed Resource Management and Scheduling for Grid Computing*, Ph.D. thesis, Monash University, Melbourne, Australia, 2002. 19, 21

[50] R. Buyya, D. Abramson, and S. Venugopal, "The Grid Economy," in *Special Issue on Grid Computing; Proceedings of the IEEE*. March 2005, vol. 93, pp. 698–714, IEEE Press, New York, USA. 19, 21

[51] C. Ernemann and R. Yahyapour, *Grid Resource Management - State of the Art and Future Trends*, Chapter Applying Economic Scheduling Methods to Grid Environments, pp. 491–506, Kluwer Academic Publishers, 2003. 19

[52] R. Yahyapour, *Design and Evaluation of Job Scheduling Strategies for Grid Computing*, Ph.D. thesis, Universität Dortmund, Informationstechnik, 2002. 19, 28, 40

[53] R. Wolski, J. S. Plank, J. Brevik, and T. Bryan, "Analyzing Market-Based Resource Allocation Strategies for the Computational Grid," *International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 258–281, Fall 2001. 20

[54] R. Raman, *Matchmaking Frameworks for Distributed Resource Management*, Ph.D. thesis, University of Wisconsin-Madison, October 2000. 20

[55] C. Dumitrescu and I. T. Foster, "GRUBER: A Grid Resource Usage SLA Broker," in *Euro-Par*, 2005, pp. 465–474. 20

[56] J. Padgett, K. Djemame, and P. M. Dew, "Grid-Based SLA Management," in *EGC*, 2005, pp. 1076–1085. 20

[57] K. M. Sim, "A survey of bargaining models for grid resource allocation," *SIGecom Exch.*, vol. 5, no. 5, pp. 22–32, 2006. 20

[58] T. Sandholm, "Service Level Agreement Requirements of an Accounting-Driven Computational Grid," Tech. Rep., Royal Institute of Technology, 2005, TRITA-NA-0533. 20

[59] C. Ernemann, V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour, "Enhanced Algorithms for Multi-Site Scheduling," in *Proceedings of the 3rd International Workshop on Grid Computing, Baltimore*. 2002, vol. 2536 of *Lecture Notes in Computer Science*, pp. 219–231, Springer. 21

[60] C. Ernemann, V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour, "On Advantages of Grid Computing for Parallel Job Scheduling," in *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID2002)*, Berlin, May 2002, pp. 39–46, IEEE Press. 21, 45

[61] A. I. D. Bucur, H. H. Mohamed, and D. H. J. Epema, "Co-Allocation in Grids: Experiences and Problems," in *in Future Generation Grids, Proc. of the Workshop on Future Generation Grids, Dagstuhl*. 2006, Springer-Verlag. 21, 69

[62] K. Czajkowski, I. T. Foster, and C. Kesselman, "Resource Co-Allocation in Computational Grids," in *Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing (HPDC-8)*, 1999, pp. 219–228. 21

[63] D. Kuo and M. Mackeown, "Advance reservation and co-allocation protocol for grid computing," in *In Proc. of the 1st IEEE International Conference on e-Science and Grid Computing, Melbourne, Australia*, 12 2005, pp. 80–89. 21

[64] O. Waeldrich, P. Wieder, and W. Ziegler, "A Meta-scheduling Service for Co-allocating Arbitrary Types of Resources," December 2005, vol. TR-0010 of *CoreGRID Technical Report*. 21

[65] T. Roeblitz and A. Reinefeld, "Co-Reservation with the Concept of Virtual Resources," in *Proceedings of the 5th IEEE International Symposium on Cluster Computing and the Grid, Cardiff, Wales, UK, May 9-12*. 2005, pp. 398–406, IEEE Computer Society. 21

[66] L. Lamport, "Paxos Made Simple," *ACM SIGACT News (Distributed Computing Column)*, vol. 32, no. 4, pp. 18–25, December 2001. 22

[67] N. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers, 1996. 22

[68] M. J. Osborne and A. Rubinstein, *Bargaining and Markets*, Academic Press, 1990. 22, 28

[69] A. R. Lomuscio, M. Wooldridge, and N. R. Jennings, "A Classification Scheme for Negotiation in Electronic Commerce," *International Journal of Group Decision and Negotiation*, vol. 12, no. 1, pp. 31–56, 2003. 22, 23

[70] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, MIT Press, 1994. 22, 28

[71] T. D. Nguyen and N. R. Jennings, "A heuristic model of concurrent bi-lateral negotiations in incomplete information settings," in *Proc. 18th Int. Joint Conf. on AI*. 2003, IEEE Computer Society, Acapulco, Mexico , 1467-1469. 23

[72] C. H. Li, J. A. Giampapa, and K. Sycara, "A Review of Research Literature on Bilateral Negotiations," Tech. Rep. CMU-RI-TR-03-41, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2003. 23

[73] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, 1947. 23

[74] S. S. Fatima, M. Wooldridge, and N. R. Jennings, "Multi-issue negotiation with deadlines," *Journal of AI Research*, vol. 27, pp. 381–417, 2006. 25

[75] P. Faratin, *Automated Service Negotiation Between Autonomous Computational Agents*, Ph.D. thesis, Department of Electronic Engineering, Queen Mary College, University of London, UK, 2000. 25, 34, 35, 47

[76] K. M. Sim, "A market-driven model for designing negotiation agents," *Computational Intelligence: special issue in Agent Technology for E-Commerce*, vol. 18, no. 4, pp. 618–637, 2002. 25

[77] K. M. Sim and S. Y. Wang, "Flexible negotiation agent with relaxed decision rules," *IEEE Transaction on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 34, no. 3, pp. 1602–1608, 06 2004. 26

[78] E.H. Gerding, D. D. B. van Bragt, and J. A. La Poutre, "Scientific approaches and techniques for negotiation: A game theoretic and artificial intelligence perspective," Tech. Rep. SEN-R0005, 2000. 26

[79] D. Zeng and K. Sycara, "Bayesian Learning in Negotiation," *International Journal of Human Computer Systems*, vol. 48, pp. 125–141, 1998. 26

[80] C. B. Excelente-Toledo and N. R. Jennings, "Using reinforcement learning to coordinate better," *Computational Intelligence*, vol. 21, no. 3, pp. 217–245, January 2005. 26

[81] V. Narayanan and N. R. Jennings, "Learning to negotiate optimally in non-stationary environments," in *Proc. 10th Int. Workshop on Cooperative Information Agents, Edinburgh, UK*, 2006, pp. 288–300. 26

[82] T. D. Nguye and N. R. Jennings, "A heuristic model of concurrent bi-lateral negotiations in incomplete information settings," in *Proceedings of International Joint Conferences on Artificial Intelligence, Mexico*, 2003, pp. 1467–1469. 26

[83] T. D. Nguyen and N. R. Jennings, "Concurrent bi-lateral negotiation in agent systems," in *Proceedings of 4th DEXA Workshop on e-Negotiations, Prague, Czech Republic.*, 2003, pp. 839–844. 26

[84] T. D. Nguyen and N. R. Jennings, "Reasoning about commitments in multiple concurrent negotiations," in *Proc. 6th Int. Conf. on E-Commerce, Delft, The Netherlands*, 2004, pp. 77–84. 27

[85] T. D. Nguyen and N. R. Jennings, "Managing commitments in multiple concurrent negotiations.," *Int J. Electronic Commerce Research and Applications*, vol. 4, no. 4, pp. 362–376, 2005. 27

[86] T. Sandholm and V. Lesser, "Leveled Commitment Contracting: A Backtracking Instrument for Multiagent Systems.," *AI Magazine*, vol. 23, no. 3, pp. 89–100, 2002. 27, 97

[87] T. Sandholm and V. Lesser, "Leveled Commitment Contracts and Strategic Breach," *Games and Economic Behavior,Special issue on AI and Economics.*, vol. 35, pp. 212–270, 2001. 27, 97

[88] C. Li, J.A. Giampapa, and K. Sycara, "Bilateral Negotiation Decisions with Uncertain Dynamic Outside Options," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Special Issue on Game-theoretic Analysis and Stochastic Simulation of Negotiation Agents*, vol. 36, no. 1, pp. 31–44, January 2006. 27

[89] H. Raiffa, *The Art and Science of Negotiation*, Harvard Universtiy Press, 1982. 27, 28, 32, 33, 35, 37, 57, 97

[90] A. Roth, *Game-Theoretic Models of Bargaining*, Chapter Toward a focal point theory of bargaining, pp. 259–268, Cambridge University Press, NY, 1985. 28

[91] T. Schelling, *The Strategy of Conflict*, Cambridge, MA: Harvard University Press, 2nd edition, 1960. 28

[92] D. Pruitt, *Negotiation Behavior*, Academic Press, 1981. 28, 35

[93] J. Tirole, *The Theory of Industrial Organization*, MIT Press, 1998, Cambridge, Massachusetts. 28

[94] S. Kraus, *Strategic Negotiation in Multi-Agent Environments*, MIT Press, Cambridge, USA, 2001. 28, 30, 92

[95] D. M. Kreps and R.Wilson, "Sequential Equilibria," *Econometrica*, vol. 50, pp. 863–894, 1982. 28

[96] A. Rubinstein, "Perfect Equilibrium in a Bargaining Model," *Econometrica*, vol. 50, pp. 97–110, 1982. 30, 68, 92

[97] K. Kurowski, J. Nabrzyski, A. Oleksiak, and J. Weglarz, *Grid Resource Management - State of the Art and Future Trends*, Chapter Multicriteria Aspects of Grid Resource Management, pp. 271–295, Kluwer Academic Publishers, 2003. 31

[98] U. Schwiegelshohn and R. Yahyapour, "Analysis of First-Come-First- Serve Parallel Job Scheduling.," in *Proceedings of the 9th SIAM Symposium on Discrete Algorithms*, January 1998, pp. 629–638. 37

[99] "Standard Workload Format," 11 2006, http://www.cs.huji.ac.il/labs/parallel /workload/index.html. 38, 40, 41, 50, 61, 73, 74

[100] D. G. Feitelson, *Workload Modeling for Computer Systems Performance Evaluation*, 2007, Version 0.9. 38

[101] B. Song, *Workload Modelling for Parallel Computers*, Ph.D. thesis, Universität Dortmund, Informationstechnik, 2006. 38

[102] D. Lifka, "The ANL/IBM SP Scheduling System," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson and L. Rudolph, Eds., pp. 295–303. Springer-Verlag, 1995, Lect. Notes Comput. Sci. vol. 949. 40, 73

[103] C. Ernemann, V. Hamscher, A. Streit, and R. Yahyapour, "On Effects of Machine Configurations on Parallel Job Scheduling in Computational Grids," in *Proceedings of the 6th Workshop Parallele Systeme und Algorithmen (PASA) in conjunction with the International Conference on Architecture of Computing Systems, ARCS*, Karlsruhe, April 2002, pp. 169–179, VDE. 40

[104] L. Kaelbling, L. Littman, and A. Moore, "Reinforcement learning: a survey," *Journal of Artificial Intelligence Research*, vol. 4, no. 1, pp. 237–285, 1996. 47

[105] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998. 47, 48

[106] C. J. C. H. Watkins, *Learning from Delayed Rewards*, Ph.D. thesis, Cambridge Univ., Cambridge England, 1989. 47

[107] K. M. Sim, "Equilibria, prudent Compromises, and the "Waiting" game.," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, no. 4, pp. 712–724, 2005. 57, 59

[108] F. Zeuthen, *Problems of Monopoly and Economic Warfare*, U.K.: Routledge and kegan-pail, 1967. 58

[109] J. Li and R. Yahyapour, "Negotiation Strategies for Grid Scheduling," in *Proceedings of the First International Conference on Grid and Pervasive Computing*. 2006, Lecture Notes in Computer Science (LNCS 3947), pp. 42–52, Springer. 59, 61, 65

[110] J. Li and R. Yahyapour, "A Strategic Negotiation Model for Grid Scheduling," *International Transactions on Systems Science and Applications*, vol. 1, no. 4, pp. 411–421, 2006. 61

[111] A. I. D. Bucur and D. H. J. Epema, "The Influence of the Structure and Sizes of Jobs on the Performance of Co-allocation," in *IPDPS '00/JSSPP '00: Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, London, UK, 2000, pp. 154–173, Springer-Verlag. 74

[112] J. Li and R. Yahyapour, "Learning-Based Negotiation Strategies for Grid Scheduling," in *To be published in Proceedings of the International Symposium on Cluster Computing and the Grid (CCGRID2006)*. 2006, pp. 576–583, IEEE Computer Society. 74, 82

[113] "Web Services Agreement Negotiation Specification," http://www.ogf.org/gf/group_info/view.php?-group=graap-wg, 2006.11. 91

[114] J. Yu and R. Buyya, "A Taxonomy of Scientific Workflow Systems for Grid Computing.," *Special Issue on Scientific Workflows, SIGMOD Record, ACM Press, New York, USA*, vol. 34, no. 3, pp. 44–49, 09 2005. 98

[115] J. Yu and R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing," *Journal of Grid Computing*, vol. 3, no. 3-.4, pp. 171–200, 09 2005. 98

[116] "Pegasus Research Project," http://pegasus.isi.edu/, 2006.11. 98

[117] "Grid Workflow Management Group," http://www.ogf.org/gf/group_info/view.php?-group=wfm-rg, 2006.11. 98