Q-Strategy: A Bidding Strategy for Market-Based Allocation of Grid Services

Nikolay Borissov[1] and Niklas Wirström[2]

[1] University of Karlsruhe, Information Management and Systems, Englerstr. 14, 76131 Karlsruhe, borissov@iism.uni-karlsruhe.de
[2] Swedish Institute of Computer Science, Box 1263, SE-164 29 Kista, Sweden
niwi@sics.se

Abstract. The application of autonomous agents by the provisioning and usage of computational services is an attractive research field. Various methods and technologies in the area of artificial intelligence, statistics and economics are playing together to achieve i) autonomic service provisioning and usage of Grid services, to invent ii) competitive bidding strategies for widely used market mechanisms and to iii) incentivize consumers and providers to use such market-based systems.
The contributions of the paper are threefold. First, we present a bidding agent framework for implementing artificial bidding agents, supporting consumers and providers in technical and economic preference elicitation as well as automated bid generation by the requesting and provisioning of Grid services. Secondly, we introduce a novel consumer-side bidding strategy, which enables a goal-oriented and strategic behavior by the generation and submission of consumer service requests and selection of provider offers. Thirdly, we evaluate and compare the Q-strategy, implemented within the presented framework, against the Truth-Telling bidding strategy in three mechanisms – a centralized CDA, a decentralized on-line machine scheduling and a FIFO-scheduling mechanisms.

Keywords: Automated Bidding, Reinforcement learning, Service Allocation, Grid Computing

1 Introduction

Distributed computing services are bundled for years to enable more efficient calculation of huge data sources and complex simulations. Prominent examples are projects like SETI@home, which bundled 418.6 TFLOPS [1] (1Mio PCs in 226 countries) of computing power, looking for extraterrestrial live. CERN's Atlas project is planned to execute complex simulations within the Large Hadron Collider, analyzing huge amounts of data in order to find novel particles and verify and analyze existing one (www.atlas.ch). Currently, the top 10 computing centers of the Top500-List (www.top500.org) offer more than 2PFLOPS of computing power. Also the industry is already using distributed services (i.e. Grid

technologies) for years and offers them already to the public. Amazon's Webservices – *Elastic Compute Cloud (EC2)*, *Simple Storage Service (S3)*, *SimpleDB* etc. – Sun's *Network.com*, Salesforce's *force.com*, Google's Apps services and *IBM Blue Cloud* are some of the prominent examples. However, studies proved that computational services are in average utilized between 10% and 35% [2]. Current workload logs (http://monalisa.caltech.edu) confirmed these results.

Grid technologies are already in use by researchers, but also the number of external developers and solution providers is rising quickly. Amazon's Web services are used by over than 160.000 developers and solution providers [3]. Studies show that businesses may reduce their total IT costs by 30%, when using such external resources [4].

The convergence of virtualization, delivery of Web service applications is going in parallel with the development of faster and more powerful computers and networks. Businesses need efficient business models for delivering their ITs and systems and frameworks, which reduce the complexity, automate software and hardware and enable an easy management of policies [5]. Market mechanisms can allocate service providers and consumers in more efficient way and thus maximizing the overall welfare. Currently, business and pricing models are not reflecting the current supply and demand of services e.g. Sun requires 1$ per CPU, Amazon has static prices for a service configuration per time unit. Moreover, common interfaces and tools are needed in order to enable a non-sophisticated access and provisioning of computational services.

This paper is structured as follows. Section 2 describes components of the *Intelligent Tools* and presents a framework, which enables implementing autonomous bidding agents. Section 3 discusses bidding strategies for automated bidding and proposes a novel consumer-side reinforcement learning bidding strategy – the Q-Strategy. In Section 4 we evaluate this strategy against the Truth-Telling strategy in three different market mechanisms. Section 5 discusses related work and Section 6 concludes this paper.

2 Automated Resource Allocation

To allocate efficiently consumers' jobs to providers' resources is a complex task, where decisions on resource provisioning and usage are executed on-line. Moreover, the wide heterogeneity of Grid services complicates the process of finding an appropriate set of resources for given consumer preferences. Since demand and supply of Grid services fluctuates in the course of time, information about current and future resource utilization and prices are often not known a-priori to the participants. In this case consumer and provider agents try to maximize their utilities by generating bids based on their valuations and historic experiences [6]. This results in strategic behavior on provider and consumer side. This paper is written in the context of the SORMA[3] project, with the focus on components and methodologies that constitutes the SORMA *Intelligent Tools*.

[3] SORMA: Self-Organizing ICT Resource Management,www.sorma-project.org

2.1 Scenario for Automated Provisioning and Usage

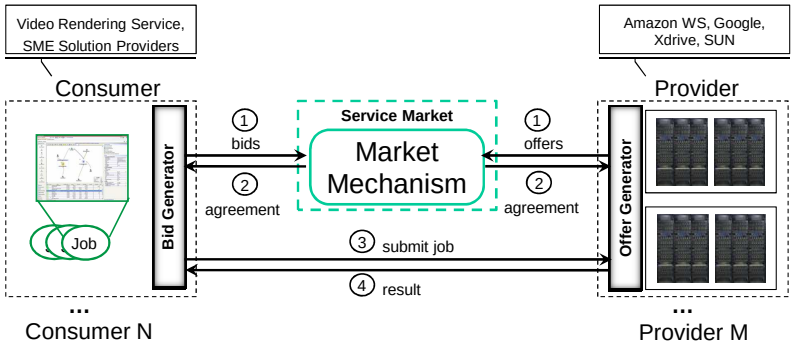

Fig. 1. Bidding Scenario

Figure 1 depicts the approach for automated provisioning and usage of Grid services. As illustrated, consumers and providers use the bid generation and respectively the offer generation component in order to generate and submit bids to the *Service Market*, which executes the target market mechanism. When an agreement has been made, both parties are informed of this and the job is submitted and executed on the allocated provider machine. Finally the result of the execution is reported back to the consumer.

2.2 Components Supporting the Automated Provisioning and Usage Process

The aim of this section is to investigate the processes and corresponding tools which enable automated bidding on provider and consumer sides. Figure 2 shows the logical architecture of the components assisting consumers and providers by the description of their technical and economic preferences as well as by the automated generation of bids and offers. In order to derive and describe his preferences, a consumer uses the *Demand Modeling* and *Economic Modeling* tools. The *Demand and Supply Modeling* components support consumer and provider in the description of requested and offered services such as CPU, memory, storage and database characteristics (see 2.4). *Economic modeling* allows consumers to describe their economic preferences e.g. specifying the valuation of the required services, the amount of time a bid is valid and the preferred bidding strategy.

Within the *Business Modeling* component, a provider has to describe his desired business model, which determines what bidding strategy to use for the generation of the offers. For example, one part of such a description is the pricing policy that specifies if the consumer has to pay for booked time-slots or for the

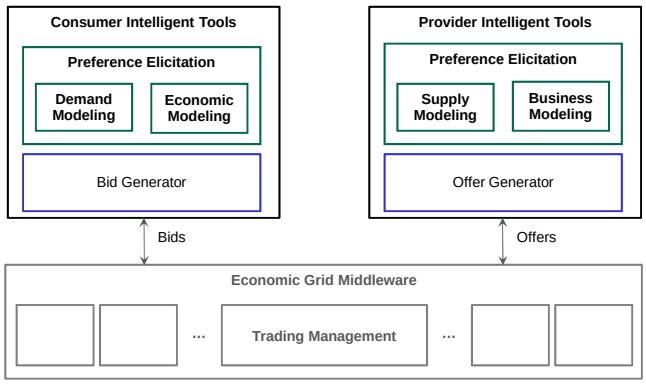

Fig. 2. Intelligent Tools for consumer and provider

actual usage. As the example indicates, the models specified by means of this component depend on the implemented market mechanism.

The *Bid and Offer Generators* are the "intelligent" components that autonomously generate and place the consumer and provider bids on the market. For this purpose it considers the specified consumer and provider preferences and the current state of the market, such as actual prices. The bid and offer generators are implemented through agents using common and novel bidding strategies and learning algorithms. The bids are submitted to the *Trading Management* component, which implements mechanisms for technical and economic matching. In the following sections we will focus on the components of the *Intelligent Tools*.

2.3 Preference Elicitation

In order to request Grid services for its application, a consumer has to make estimation regarding his preferences on technical service specification, QoS and his willingness to pay. On the other side, a provider has to make price estimation for its offered services. These consumer and provider preferences for a specific application can be either static or dynamic. Static information is collected once, and can be manually provided by the consumer or provider each time a service has to be acquired or offered. The static information may also be stored in databases enabling intelligent tools to use this information for predicting requirements of applications and services with similar properties. However, the requirements of a given application are often subject to change as technology evolves e.g. consumers' desired quality of streamed video might increase as their Internet bandwidth increases. It is thus desirable to dynamically adapt the service requirements. In [7] the authors specify a model for *Job Valuation Estimation* using evolutionary techniques. The presented approach is based on the assumption that a consumer who wants to buy a set of services does not

generally know his exact valuation for them, but has only a rough estimate of his true valuation. Thus, he decides whether to accept the offer, or to continue his search and look for alternative offers.

2.4 Demand and Supply Modeling Components

Both, the *Demand and Supply Modeling* components support consumers and providers on editing their estimated preferences – technical requirements on services, such as CPU, Memory and Storage, QoS and price. The component implements a GUI for entering the consumer or provider preferences and generates a XML output in form of predefined service description language such as *Job Submission and Description Language* [8]. The main parts of this component are the *User Interface*, which allows the input of technical service specifications on consumer and supplier side and a *Service Description Language* for expressing the service specification, traded on the market. To allow different levels of abstraction and granularity, the service needs to be technically specified in terms of its grounding hardware and the required software environment. Together with the technical specification, comes the specification of several non-functional service properties like QoS. Further sections like economic parameters, job-specific parameters or possible inter-job dependencies complete the service specification.

2.5 Bid and Offer Generator

In the *Service Market* scenario each consumer and provider is configuring and using the intelligent tools i.e. bidding agents in order to use or provide services with the objective to maximize its own utility.

The bid and offer generator components are implemented within a *Bidding Agent Framework*, which core classes and relations are illustrated in Figure 3. The framework defines and implements core processes, which enable "automated bidding behavior". The left part of the framework handles the technical description and QoS of the offered or requested services, received from the demand and supply modeling component (see section 2.4). The right part, handles the bidding strategy i.e. economic preferences received from the economic and business modeling component. According to the framework, a bidding *strategy* implements the bidding behavior of the bidding agent, e.g. how, when and what to bid. For this purpose it adopts learning algorithms to learn from earlier actions and predict future rewards by selecting a particular price for a given service description. A strategy profile can be configured with policies, which are defined in a rule description language and executed within a rule engine. *Policies* in our case are *scoring* and *pricing* functions, which are defined externally to the implementation and thus enable a flexible modification. Through the scoring function, the participant specifies the overall objectives as a mathematical function that is to be maximized by its bidding agent. For a job j, the pricing policy enables a static specification of a valuation v_j or price calculation function, which is used by the bidding strategy to calculate the bid $\tilde{v}_j \leq v_j$, which is reported to the Service Market. And last step is to generate the final bid message containing the

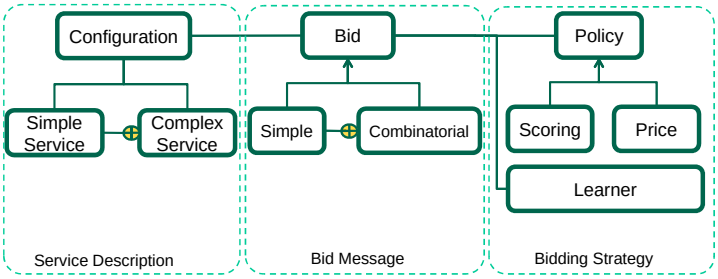

Fig. 3. Bidding Agent Framework

provided or requested technical service description with the economic preferences like bid, duration and time validity.

3 Bidding Strategies

To implement a strategic behavior on the consumer side, we implemented consumer agents using two rational bidding strategies, a non-adaptable *Truth-Telling*, where the generated bid do not depend on past experience, and adaptable *Reinforcement Learning* bidding strategy, named Q-Strategy. The reinforcement learning bidding strategy is implemented through the Q-Learning algorithm with epsilon-greedy selection strategy (section 3.3).

Generally in this paper, we do not consider strategic behavior on provider side, where provider agents adopt some bidding strategies to generate the price for their offered services. However, we implemented a *Continuous Double Action* (CDA) (see section 4), which requires strategic behavior on both sides, where consumer and provider are continuously matched based on their bids. Therefore we present and discuss the Zero Intelligence Plus Strategy [9] on provider side for the evaluation of the CDA mechanism.

3.1 Truth Telling Strategy

In the model of [10], agents do not remember the outcomes of earlier market interactions, but are somewhat "myopic" in the sense that they only consider the current situation. As shown for the model in [10], without knowledge about the future, at time $\tilde{r}_j$ it is a utility maximizing strategy s_j for agent $j \in J$ to report its true type t_j to the system and to choose the machine i which maximizes $\hat{u}_j(i|s_{-j}, \tilde{t}_j, t_j)$.

The *Truth-Telling* bidding strategy places a bid price, which equals the provider's or consumer's valuation for a certain service. Although a simple strategy, truth-telling is essential in case of strategy-proof mechanisms, where in such mechanisms this strategy guarantees to obtain optimal payoffs, no matter what strategies are adopted by the others. However, in budget-balanced double-auction mechanisms, this strategy is not dominant [11].

3.2 Zero Intelligence Plus Strategy

Zero-Intelligence Plus (ZIP) agents are widely explored and become a popular benchmark for agents trading on continuous double auctions [9].In [12], the author showed that zero intelligence agent strategy [13] is not enough, since the bids are uniform generated between a given interval and not depend on current market information or past bids. They introduced a new type of agent, the ZIP agent, which relies on a reinforcement learning method to learn the price of a particular market.

Central to the ZIP agent is the rule for updating the profit margin μ. For a buyer agent, this is the difference between its valuation v (i.e. maximum willingness to pay) and the generated bid b. The relationship between profit margin, generated bid and the valuation is like following:

$$b_i = (1 + \mu_i)v \tag{1}$$

The rules for raising and lowering their profit margins are based on whether the last shout was a bid or an offer, and whether the agent is active (i.e. has service to sell, or service to buy). In [9], experiments show that ZIP agents perform better than (non-expert) human traders on CDA markets. Simulations show that adopting a ZIP strategy in markets dominated by other kinds of agents, results in an increased profit when the dominating agents are of the kind ZI, Kaplan or Gjerstad and Dickhaut(GD)[14] agents, and a small decrease if the market is dominated by MGD agents[15]. In most experiments in the literature[12, 15, 9], the ZIP agent receives information of all bids and offers, whether they are accepted or not. There are, however, experiments showing that ZIP based agents perform well also in single sided sealed bid auctions where the agents only receive information on the winning bids, and the price, the winner has to pay [16].

In [17], genetic algorithms (GA) are used to find optimal values for the initialization parameters. This is done by using eight dimensional genotypes representing the upper bounds for the distributions from which there are drawn and upper and lower bounds for three different uniform distributions from which a value for each of the variables are drawn independently for each agent in the market. The agents in these markets have come to be called ZIP8 [18] because of length of the genotype. In a later paper [18], Cliff presents a revised version of the same approach. Here, the genotypes have the length 60, and consequently, the agents in these markets are referred to ZIP60 agents. The increase in length from 8 to 60 is due to that new parameters, making it possible for the GA algorithm to find different sets of distributions for buyers and sellers, and also parameters for each of the three cases in which the profit margin is updated.

The ZIP strategy is a state-of-the-art in CDA markets, where the agents adapt their bids based on public bid signals. It performs well by high demand and supply and converges quickly to equilibrium price in CDA markets. This strategy is mainly evaluated and designed for CDA markets and assumes that all bids submitted to the market are public to the agents. In the following section we introduce a novel bidding strategy – Q-Strategy for consumers, which does not require public information.

3.3 The Q-Strategy

Our aim is to develop rational agents with learning capabilities, which may strategically misreport about their true valuation based on previous experiences. We refer to these strategies as "rational response strategies". The Q-Strategy

Algorithm 1 Q-Strategy: Bid Generation Rule

Require: $economicpreferencesofthejob$
1: **if** $\epsilon < Stochastic.random(0,1)$ **then**
2: $//Explore:$
3: $scale \in (0,1)$
4: $price = Stochastic.random(scale * job.getValuation(), job.getValuation())$
5: **else**
6: $//Exploit:$
7: $state = State.getState(job)$
8: $action = qLearner.bestAction(\text{state})$
9: **if** $action! = nil$ **then**
10: $price = action.getBidPrice()$
11: **else**
12: $price = job.getValuation()$
13: **end if**
14: **end if**

consists of two algorithms (see Algorithm 1 and 2). The first algorithm describes the case where an agent generates a bid (or offer) for a given configuration of services it wants to buy (or sell). The second algorithm applies to the case where an agent receives a number of offers for a given configuration of services, and has to select one of them to buy.

Both algorithms are based on a reinforcement learning approach – Q-Learning [19] with a ϵ-greedy selection policy [20, 21]. Using this policy, the agent explores the environment with a probability of ϵ, by selecting an action randomly, and exploits its obtained knowledge with probability of $1 - \epsilon$, by selecting an action that has been beneficial in the past.

We use the following notation:

- Each job j has a type $t_j = \{S_j, r_j, d_j, v_j\}$, where S_j is the specification of technical requirements for an application or a job e.g. number of CPUs, storage units and database service, r_j represents the release time of a job, d_j the requested duration and v_j its valuation. Furthermore, each job type is associated to a class of common job types $t_j \in T_j$, where the class parameters S_j, r_j, d_j and v_j are learned for all "similar" job-types.
- S is a finite discrete set of states, where each state s is defined by a tuple $\{S, d, v\}$, such that an agent is in state $s = \{S_j, d_j, v_j\}$ if it is to bid for a job with a specification of technical requirements S_j, duration d_j and valuation v_j.

- A is a repertoire of possible actions, where, in the context of this paper each action a represents the assignment of a specific bid price.
- $Q(s, a)$ denotes the expected value of being in state s and taking action a.
- ρ is a mapping from stimuli observed, caused by an action, to the set of real numbers. Here, we use $\rho = -v_j C_j - \pi_j$, where C_j is the time-span between creation and completion of job j, and π_j is the price paid for it.

In other words, the objective is to learn the function $Q(s, a)$, so that, given any job with a specific technical requirements, duration and valuation, a price $\tilde{v} \leq v$ can be selected so that the utility is maximized. However, due to the sizes of the state and action spaces, and the fact that the environment in which the agent operates is continuously changing, on this stage only a rough estimate of the Q-function is feasible. The dynamic adaptation of the learning rate will influence the probability to be either in the "exploration phase" or "exploitation phase" and thus the reaction of the changing conditions. Thus, the $Q(s, a)$-function is learned with the time, it represents a kind of "aggregated" experience for a given class of jobs with similar technical requirements, durations as well as valuations. Currently, we define similar jobs-types as equal regarding technical parameter specifications, durations or valuations and do not yet introduce fuzzy rules to consider some deviations on these.

As stated earlier, learning is made through exploration of the environment. After finishing a job, the Q-function, is updated with the new information according to the *Q-Learning update rule*:

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha_t(s_t, a_t)[\rho_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \qquad (2)$$

Here, s_t is the state defined by the duration and valuation of the job that the agent bids for at time t, a_t is the action selected at time t, ρ_t is the received utility of the job. The learning rate $\alpha_t \in [0, 1]$ determines how much weight we give to newly observed rewards. A high value of α_t results in that high importance is given to new information, while a low value leads to that the Q-function is updated using small steps. $\alpha_t = 0$ means no learning at all. The discounting factor γ defines how much expected future rewards affect current decisions. Low ($\gamma \to 0$) implies higher attention to immediate rewards. Higher ($\gamma \to 1$) implies orientation on future rewards, where agents may be willing to trade short-term loss for long-term gain. In Algorithm 1, during exploration, the bid is randomly generated within the interval $p^s \in [s * v_j, v_j]$ with $s \in (0, 1)$. During exploitation, the bid is retrieved from the Q-Table, the "best" bid that achieved the highest average payoff in the past. In Algorithm 2, during exploration, the strategy selects the "best" (utility maximizing) service provider offer, for which there is no stored information in the Q-Table. During exploitation it selects the best offer, which maximizes its utility.

The introduced Q-Strategy seems to be rational in the sense to learn the "optimal" consumer bids for given consumer preferences. In [19], the author show that Q-Learning converges to optimal action values, which is also assumed for the generated bids, modeled as actions. The jobs are assigned to job-classes

Algorithm 2 Q-Strategy: Offer Selection Rule

Require: $economicpreferencesofthejob; provideroffers$
1: **if** $\epsilon < Stochastic.random(0,1)$ **then**
2: $//Explore:$
3: $offer = bestOfferForProviderNotStoredInQTable(job, offers);$
4: **else**
5: $//Exploit:$
6: $offer = myopicBestResponse(state, offers);$
7: **end if**

to learn the bids and preferences for "similar" jobs and converge quicker to "optimal" bids. The main advantage of this strategy is that the agents does not require public information i.e. other agent bids to adapt. A common drawback of reward-based learning is that to learn the optimal bid price needs "training time", thus in the worse case this strategy can perform worse at the beginning, but with the time converge to "optimal" actions.

4 Evaluation

Auction and strategy selection are closely connected in the sense that a given choice of strategy should affect the choice of auction, and vice versa. For example, some bidding strategies perform well in a Continuous Double Auction (CDA), but not in a Dutch auction. This also implies that the choice of auction to participate in depends on the available strategies. Other factors to take into account in auction selection are the market rules, transaction costs, and the current and average prices in the different auctions.

In this section, we evaluate the Truth-Telling and Q-Learning strategies for three different types of market mechanisms for allocating Grid services. The first market mechanism is a decentralized on-line machine scheduling mechanism, called Decentralized Local Greedy Mechanism (DLGM) [10]. The second one is a centralized continuous double auction (CDA) [15]. The third mechanism we implemented is FIFO (First-In-First-Out), a state-of-the-art scheduling mechanism, representing the class of technical schedulers and serves as a baseline mechanism for comparing the results. The market mechanisms are implemented within the *SORMA Trading Management* component. The simulation is run on a light version of this component.

As a measure we use the average utility per job received by the consumers. In the Truth-Telling scenario, this is the same as measuring the common wealth for this particular strategy, since all players have the same strategy. In the case of the Q-Learning strategy, however, this is not equivalent, since the behaviors of the players diverge as the players observe different information.

In the case of the decentralized DLGM mechanism, each time a job arrives on the consumer side, his bidding agent generates a bid in form of a job type $t_j = \{S_j, r_j, d_j, v_j\}$ (see Section 3) and report this to all providers. Based on this bid, each provider reports back a tentative completion time and tentative

price for each of its machines. When sufficiently many provider offers have been collected, the consumer can decide, based on his utility function $\rho = -v_j C_j - \pi_j$ [10], sections 2.5 and 3.3, which offer to choose. The providers in the DLGM market do not behave strategically and do not request compensation for the use of their services. The payments are divided only among the consumer agents for compensating the displaced jobs.

In the centralized CDA, consumers and providers submit bids and offers consisting of only a price per time unit, and are matched based on this information. Providers act strategically, trying to achieve as much money as possible for their services. To calculate the price of their bids, they use a ZIP (Zero Intelligence Plus) agent [12].

In the FIFO mechanism the agents' submit their jobs to a provider machine, which offered the shortest completion time, where the jobs are scheduled and executed through the FIFO principle. In the following section we describe the evaluation settings and simulation results.

4.1 Evaluation Setting

For each market – DLGM, CDA and FIFO, and each strategy – Q-Learning and Truth-Telling, we simulated four different scenarios described by settings 1 through 4 in Table 1. In each scenario there are 50 consumers and 50 providers (each controlling a single machine). In each of the first three settings, the rate of which jobs come in on the consumer side is determined by a Poisson process. To increase the competition in the market, we successively increased the mean λ of the Poisson process from λ=.1 (setting 1) to λ=.5 (setting 3). The amount of jobs for these settings is a direct consequence of these values. For these settings, the duration of each job is drawn from the normal distribution with a mean value of 5 hours and a variance s^2 of 3.

The fourth setting is based on the logs of a real workload at the LPC (Laboratoire de Physique Corpusculaire) cluster which is a part of the EGEE Grid environment, and located in Clermont-Ferrand, France [22]. The log contains 244,821 jobs that were sent to the nodes during a period of 10 months starting from August 2004 through May 2005. We have, however, only extracted jobs with duration between one and 24 hours. The LPC log was chosen because it contains a large variety of jobs with different run-times, numbers of used CPUs, and varying submit and start times.

Table 1. Simulation settings

Setting	Arrival Rate	Duration	# Jobs	# Consumers	# Providers
1	$Poisson(0.1)$	$\max(1, N(5,3))$	751	50	50
2	$Poisson(0.3)$	$\max(1, N(5,3))$	1502	50	50
3	$Poisson(0.5)$	$\max(1, N(5,3))$	3004	50	50
4	As in LPC-Log	As in LPC-Log	105,578	50	50

4.2 Simulation results

The results of the simulations are summarized in Table 2. Each line in the table represents the evaluation of one strategy for one setting. The first two columns represent the setting used (corresponding to those of Table 1) and the strategy evaluated. The next two columns represent the average utility μ per job achieved – here $\rho = -v_j C_j - \pi_j$ [10] – as well as the standard deviation σ of job budget and actual payment in the DLGM market, and the last four columns represent the results for the CDA and FIFO in the same way.

The results show that the *Truth-Telling* strategy achieves the highest utility for all four settings in both DLGM and CDA markets except by the FIFO mechanism. FIFO achieved "slightly better" utility using Q-Strategy against Truth-Telling, although it should be exact the same, because of the FIFO behavior and the same on-line decisions in both strategy cases. This slight deviation can be explained due to contorted decision time of both strategies, where by the Truth-Telling strategy the bid corresponds the true valuation and by the Q-Strategy is determined based on its ϵ-greedy phase (see section 3.3). The *Q-Strategy* repro-

Table 2. Simulation results

Setting	Strategy	$DLGM\mu$	$DLGM_\sigma$	CDA_μ	CDA_σ	$FIFO_\mu$	$FIFO_\sigma$
1	Truth-Telling	$-110,48$	$272,37$	$-7,92 * 10^4$	95,33	-139,78	8,13
1	Q-Strategy	$-174,74$	$257,54$	$-10,33 * 10^4$	93,56	-134,82	8,13
2	Truth-Telling	$-212,66$	$285,16$	$-11,95 * 10^4$	94,13	-276,92	8.08
2	Q-Strategy	$-392,42$	$265,96$	$-14,63 * 10^4$	93,30	-265,30	8.08
3	Truth-Telling	$-403,58$	$286,43$	$-7,89 * 10^4$	86,74	-549,12	8.06
3	Q-Strategy	$-901,18$	$265,24$	$-23,22 * 10^4$	90,77	-524,74	8.06
4	Truth-Telling	-1104	$647,27$	$-9,91 * 10^4$	391,97	-4532,0	400.93
4	Q-Strategy	-1172	$580,68$	$-11,04 * 10^4$	313,69	-4444.0	400.93

duces a strategic behavior by the generation of the bids. More specifically, we assume that rational agents have an incentive to understate their true price in relation to their valuation. Due to the fact that they understate their true price the achieved utility is lower than by the *Truth-Telling* strategy. The simulation results showed that bidding truthfully in both mechanisms can only increase your utility. Understating the truthful valuation in lower bid results in a poorer "job priority" p_j/d_j by DLGM and this job can be displaced by other jobs which have higher priority. As listed in table 2 the DLGM mechanism outperform the FIFO mechanism in all four settings using the Truth-Telling bidding strategy and achieves the highest common wealth for all participating consumer agents. Using the Q-Strategy all agents reveal their true valuation, which leads to worse common wealth against FIFO. When the overall number of jobs is high (setting 4), agents submit and execute continuously jobs for a longer period of time i.e. the agents have enough "training time" to "learn" the bids, in DLGM, the setting with agents using the Q-Strategy has "slightly" lower utility than the setting

with agents reporting truthfully (Truth-Telling). This result is also strength by the fact that in this setting the Q-Strategy outperforms four times the common wealth of the FIFO mechanism.

By the CDA mechanism, the price depends on the current demand and supply, bidding a lower price instead of the truth valuation increases the risk of no allocation by the mechanism. Like in the DLGM market mechanism, the Truth-Telling strategy in the CDA market mechanism achieves higher average utility than the Q-Strategy. However, by the specified CDA market mechanism, the matching is based only on the price without considering the "priority" of a job as with DLGM, and thus achieves very low utility compared to DLGM. The origin of this can be searched in the CDA mechanism itself. First, each agent – provider and consumer – receives all the bids of the other agents as public information and based on this they adapt their bid/offer through the implemented bidding strategy. The CDA-provider agents are also acting strategically and adapting their offered price based on the received public information. Thus, the matching is based on the price resulting from the demand and supply and not on the "job priority" as with DLGM. Secondly, the CDA-provider machine agents do not maintain a priority queue of the submitted job bids and by an allocation the job is immediately submitted and executed on the provider machine. A provider submits an offer as soon as he becomes idle. Thus each time the agents are competing by adapting their job bids based on the used strategy.

4.3 Convergence of the Q-Strategy

Furthermore we investigate the bid convergence of the Q-Strategy itself using the real workload data of setting 4 (105.578 jobs), because of the high number of jobs. Figure 4 shows the bid generation in both phases – exploration and exploitation – during the simulation time of setting 4 and explicitly the corresponding bid selection in the exploitation phase of eight different consumers for eight selected particular classes of jobs. The selection of the consumers and their jobs is based on statistical analysis of the output data, where we selected classes of jobs of different consumers, which have a statistically high number of generated bids. The minimum number of generated bids per job-class is 1, the maximal 49 and the average 12.

Each graph shows the valuation of a particular job class – e.g. consumer agent 8, job class of jobs with valuation of 86 cents and requested duration of 2 hours – as a horizontal line, the development of the bid over the time and the convergence trend of the bid displays the dotted line. As displayed on the first left graph, the generated bids are fluctuating based on the ϵ-greedy phase of the Q-Strategy. The neighbor graph to the right side shows the extraction of the bids, selected on the exploitation phase. An interesting result is that for some cases of jobs with lower "job priority"–p_j/d_j, bids does not converge to the true valuation and for jobs with higher priority, bids converge to the truth-valuation of the specific job type.

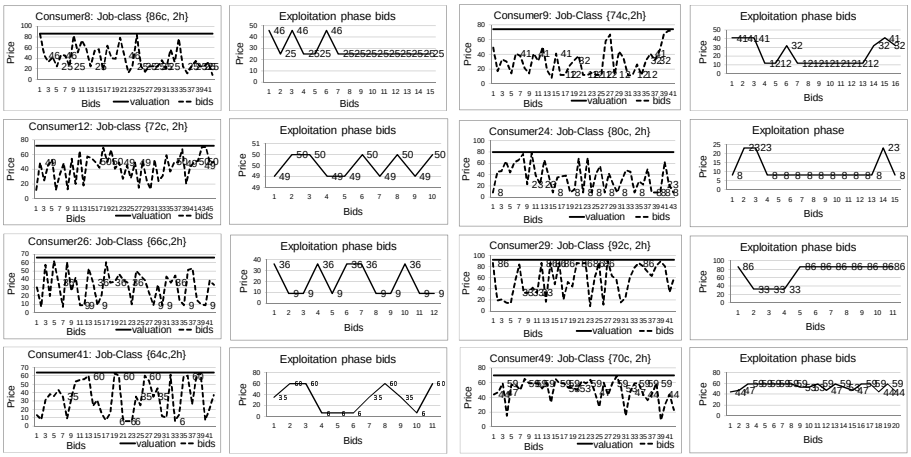

Fig. 4. Convergence of the bids using Q-Strategy

5 Related Work

Economic models for resource scheduling are widely explored in the literature [23–27]. According to their mode of allocation, scheduling mechanisms can be distinguished into mechanisms which execute periodically–"offline mechanisms", and mechanisms which execute continuously–"online mechanisms". Market mechanisms e.g. Vickrey, English, Dutch, and Double Auctions [28] as well as combinatorial mechanisms [29, 30] involve a scheduling problem that is NP-hard. The computational complexity of such mechanisms drives researchers and practitioners to look at simpler allocation models which can be adapted to real-world scenarios and requirements. Prominent examples are proportional share mechanisms [31], where the users receive a share of the computer resource proportional to their valuation's fraction of the overall valuation across all users. A related on-line mechanism is the so-called pay-as-bid mechanism proposed in [32]. In pay-as-bid mechanisms, the agents submit only a single real value as bids which at the same time make the agents' final payments.

Preference elicitation deals with extraction of user's preferences for different combinations of resource configurations and prices. The aim of this methodology is to find an "optimal" choice of configuration, without explicitly presenting all possible choices. Two important approaches for job preference estimation are discussed in the literature – *Conjoint Analysis* and *Analytical Hierarchy Process* [7, 33, 34]. Conjoint analysis estimates the user's value for a certain attribute level by performing regression analysis on the user's feedback to the presented attribute profiles. In contrast to the conjoint analysis method which aims at determining the value of a certain attribute, the analytical hierarchy process tries to determine the relative importance of a certain attribute among a set of attributes [35].

The field of autonomous bidding is explored by many researches. [36] gives an overview of the various agents and their strategies taken place in the *Trading Agent Competition*. The literature described trading agents in stock markets [37], supply chain management [Pardoe and Stone 2007] and in various market mechanisms [38, 39] Since agents are self-interested, they aim to implement a strategic behavior in order to maximize their utilities. In this context the mechanism design and auction literature investigated various bidding strategies for market-based scheduling [40, 11, 41, 42]. Phelps elaborated co-evolution algorithms to learn the strategy space for autonomous bidding by the allocation of resources in market mechanisms. In his thesis he classified bidding strategies into non-adaptive – *Truth Telling*, *Equilibrium-Price* and *Zero Intelligence* strategy (see section 3.2) – and adaptive strategies – *Zero-Intelligence Plus*(see section 3.2), *Kaplan's Sniping Strategy*, *Gjerstad-Dickhaut* and *Reinforcement-learning*[20]. [14] GD-agents store history information about the submitted bids and use belief function for the price estimation. FL-agents [39] use fuzzy logic to generate a bid or offer based on a base price, which is a median of previous prices. Risk-Based agents [38] perform prediction of expected utility loss resulting from missing out on a transaction. Kaplan agents [43] define strategic conditions ("juicy offer", "small spread" and "timeout") under which a bid is generated and submitted. A comparison between common bidding strategy is evaluated by [9, 37].

The management of computational resource centers as well as scalable applications often needs to be automated. "Intelligent" agents can take automated decisions based on current resource utilizations and the preferences of the participants. Modeling such decision making behaviors is one of the central problems in AI research. A traditional approaches in the this case are often based on Belief-Desire-Intention[44] models, where beliefs correspond to information that the agent has about the environment. Desires represent "options" available to the agent to different possible actions an agent may choose. Intentions represent states that the agent has chosen and committed to use. In this case, the agent's reasoning involves repeatedly updating beliefs from information in the environment, deciding what actions are available, examining these options to determine new intentions and acting on the basis of these intentions. Based on the preferences of the participants, the agents handle with the aim maximizing their expected utility. This led to the adoption of numerical methods based on dynamic programming such as reinforcement learning, in which the symbolic concept of a goal is replaced by a numerical reward value.

The AI literature introduces three main approaches for learning – supervised, unsupervised, and reward-based learning. These methods are distinguished by what kind of feedback the critic provides to the learner. In supervised learning, the critic provides the correct output. In unsupervised learning, no feedback is provided at all. In reward-based learning, the critic provides a quality assessment (the "reward") of the learner's output. A wide summary of common learning algorithms and decision rules are presented by [45–47, 37, 15, 48, 49]

6 Conclusions and Outlook

In this paper we have described consumer and provider components supporting the automated bidding process. We presented a framework for automated bidding, which offers a methodology and core concepts for implementing configurable bidding agents. We introduced the Q-Strategy as novel consumer bidding strategy, which implements a rational strategic behavior by the provisioning and requesting of Grid services, and evaluated it against the *Truth-Telling* bidding strategy in three different mechanisms. We show that the Q-Strategy tends to converge to optimal action values.

A common drawback of reinforcement learning algorithms is that they need some time to learn the environment and start to converge to an optimal action. To evaluate the properties of the Q-Strategy we need further research and simulations with different simulation settings e.g. mixed scenarios and in further mechanisms e.g. proportional-share [31, 50, 50] and pay-as-bid [51] as well as a comparison against state-of-the art bidding strategies like *ZIP* (also on consumer side), *GD* and *Kaplan* agents. Moreover, we investigated strategic behavior on consumer side, where truth telling is supposed to be an optimal bidding strategy in the sense of maximizing the consumer's utility. Next steps will conclude the investigation of strategic behavior on provider side by extending the DLGM mechanism with payments for the service usage. Bidding strategies like Q-Strategy will be also introduced on provider side. In this case the truth telling strategy could be not optimal.

References

1. Cirne, W., Brasileiro, F., Andrade, N., Costa, L., Andrade, A., Novaes, R., Mowbray, M.: Labs of the world, unite!!! Journal of Grid Computing **4**(3) (2006)
2. Andrzejak, A., Arlitt, M., Rolia, J.: Bounding the resource savings of utility computing models. (2002)
3. Barr, J.: Successful web based business web sites, amazon web services. WWW2006 (2006)
4. Minoli, D.: A networking approach to grid computing. Wiley Hoboken, NJ (2005)
5. Knights, M.: Grid computing can power your business. Distributed Computing — Computer Hardware, ComputerWeekly (13 Jul 2007)
6. Smith, W., Foster, I., Taylor, V.: Predicting application run times using historical information. Lecture Notes in Computer Science **1459** (1998)
7. Stoesser, J., Neumann, D.: A model of preference elicitation for distributed market-based resource allocation. Working paper, University of Karlsruhe (TH) (2008)
8. Anjomshoaa, A., Brisard, F., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D. Savva, A.: Job Submission Description Language (JSDL) Specification, Version 1.0. Job Submission Description Language WG. (2005)
9. Das, R., Hanson, J., Kephart, J., Tesauro, G.: Agent-human interactions in the continuous double auction. Proceedings of the International Joint Conference on Artificial Intelligence **26** (2001)
10. Heydenreich, B., Müller, R., Uetz, M.: Decentralization and Mechanism Design for Online Machine Scheduling. METEOR, Maastricht research school of Economics of TEchnology and ORganizations (2006)

11. Phelps, S.: Evolutionary mechanism design. Ph.D. Thesis (July 2007)
12. Cliff, D.: Minimal-intelligence agents for bargaining behaviors in market-based environments. TechnicalReport, Hewlett Packard Labs (1997)
13. Gode, D., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. The Journal of Political Economy **101**(1) (1993) 119–137
14. Gjerstad, S., Dickhaut, J.: Price formation in double auctions. Games and Economic Behavior **22**(1) (1998) 1–29
15. Tesauro, G., Das, R.: High-performance bidding agents for the continuous double auction. Proceedings of the 3rd ACM conference on Electronic Commerce (2001)
16. Bagnall, A.J., Toft, I.: Autonomous adaptive agents for single seller sealed bid auctions. Journal of Autonomous Agents and Multi-Agent Systems (2005)
17. Cliff, D.: Explorations in evolutionary design of online auction market mechanisms. Electronic Commerce Research and Applications **2**(2) (2003) 162–175
18. Cliff, D.: Zip60: an enhanced variant of the zip trading algorithm. Proceedings of the The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (2006)
19. Watkins, C., Dayan, P.: Q-learning. Machine Learning **8**(3) (1992) 279–292
20. Kaelbling, L., Littman, M., Moore, A.: Reinforcement learning: A survey. Arxiv preprint cs.AI/9605103 (1996)
21. Whiteson, S., Stone, P.: On-line evolutionary computation for reinforcement learning in stochastic domains. Proceedings of the 8th annual conference on Genetic and evolutionary computation (2006) 1577–1584
22. Medernach, E., des Cezeaux, C.: Workload analysis of a cluster in a grid environment. Job Scheduling Strategies for Parallel Processing: 11th International Workshop, JSSPP 2005, Cambridge, MA, USA, June 19, 2005 (2005)
23. Nassif, L., Nogueira, J., de Andrade, F.: Distributed resource selection in grid using decision theory. Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (2007) 327–334
24. Buyya, R.: Economic-based Distributed Resource Management and Scheduling for Grid Computing. PhD thesis, Monash University (2002)
25. Parkes, D., Singh, S., Yanovsky, D.: Approximately efficient online mechanism design. Proc. 18th Annual Conf. on Neural Information Processing Systems (2004)
26. Wolski, R., Plank, J., Brevik, J., Bryan, T.: Analyzing market-based resource allocation strategies for the computational grid. International Journal of High Performance Computing Applications **15**(3) (2001) 258
27. Walsh, W., Wellman, M., Wurman, P., MacKie-Mason, J.: Some economics of market-based distributed scheduling. Distributed Computing Systems, 1998. Proceedings. 18th International Conference on (1998) 612–621
28. Grosu, D., Das, A.: Auctioning resources in grids:model and protocols. Concurrency and Computation **18**(15) (2006)
29. Schnizler, B., Neumann, D., Veit, D., Weinhardt, C.: Trading grid services-a multi-attribute combinatorial approach. European Journal of Operational Research, forthcoming (2006)
30. Bapna, R., Das, S., Garfinkel, R., Stallaert, J.: A market design for grid computing. (2005)
31. Lai, K., Rasmusson, L., Adar, E., Zhang, L., Huberman, B.: Tycoon: An implementation of a distributed, market-based resource allocation system. Multiagent and Grid Systems **1**(3) (2005) 169–182

32. Sanghavi, S., Hajek, B.: Optimal allocation of a divisible good to strategic buyers. 43rd IEEE Conference on Decision and Control-CDC (2004)
33. Luce, R., Tukey, J.: Simultaneous conjoint measurement: A new type of fundamental measurement. Journal of Mathematical Psychology **1**(1) (1964) 1–27
34. Green, P., Rao, V.: Conjoint measurement for quantifying judgmental data. Journal of Marketing Research **8**(3) (1971) 355–363
35. Saaty, T.: Axiomatic foundation of the analytic hierarchy process. Management Science **32**(7) (1986) 841–855
36. Wellman, M., Greenwald, A., Stone, P.: Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition. MIT Press (2007)
37. Sherstov, A., Stone, P.: Three automated stock-trading agents: Acomparative study. Agent-mediated Electronic Commerce VI: Theories for and Engineering of Distributed Mechanisms and Systems: AAMAS 2004 Workshop, AMEC 2004, New York, NY, USA, July 19, 2004: Revised Selected Papers (2006)
38. Vytelingum, P., Dash, R., David, E., Jennings, N.: A risk-based bidding strategy for continuous double auctions. Proc. 16th European Conference on Artificial Intelligence (2004) 79–83
39. He, M., Leung, H., Jennings, N.: A fuzzy-logic based bidding strategy for autonomous agents in continuous double auctions. IEEE Transactions on Knowledge and Data Engineering **15**(6) (2003) 1345–1363
40. Reeves, D., Wellman, M., MacKie-Mason, J., Osepayshvili, A.: Exploring bidding strategies for market-based scheduling. Decision Support Systems **39**(1) (2005)
41. Li, J., Yahyapour, R.: Learning-based negotiation strategies for grid scheduling. Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)-Volume 00 (2006) 576–583
42. Li, J., Yahyapour, R.: A strategic negotiation model for grid scheduling. Journal International Transactions on Systems Science and Applications (2006) 411–420
43. Kaplan, S., Weisbach, M.: The success of acquisitions: Evidence from divestitures. The Journal of Finance **47**(1) (1992) 107–138
44. Haddadi, A.: Belief-desire-intention agent architectures. Foundations of Distributed Artificial Intelligence (1996)
45. Stone, P.: Learning and multiagent reasoning for autonomous agents. In: The 20th International Joint Conference on Artificial Intelligence. (January 2007) 13–30
46. van den Herik, H.J., Hennes, D., Kaisers, M., Tuyls, K., Verbeeck, K.: Multi-agent learning dynamics: A survey. In Klusch, M., Hindriks, K.V., Papazoglou, M.P., Sterling, L., eds.: CIA. Volume 4676 of Lecture Notes in Computer Science., Springer (2007) 36–56
47. Erev, I., Roth, A.: Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. The American Economic Review **88**(4) (1998) 848–881
48. Shoham, Y., Powers, R., Grenager, T.: If multi-agent learning is the answer, what is the question? Artificial Intelligence **171**(7) (2007) 365–377
49. Panait, L., Luke, S.: Cooperative multi-agent learning: The state of the art. Autonomous Agents and Multi-Agent Systems **11**(3) (2005) 387–434
50. Stoica, I., Abdel-Wahab, H., Jeffay, K., Baruah, S., Gehrke, J., Plaxton, C.: A proportional share resource allocation algorithm for real-time, time-shared systems. Proceedings of the 17th IEEE Real-Time Systems Symposium (1996)
51. Sanghavi, S., Hajek, B.: Optimal allocation of a divisible good to strategic buyers. 43rd IEEE Conference on Decision and Control-CDC (2004)