2009

A model of preference elicitation: The case of distributed resource allocation

Jochen Stößer
Institute of Information Systems and Management, Universität Karlsruhe, stoesser@iism.uni-karlsruhe.de

Dirk Neumann
Institute of Information Systems and Management, Universität Karlsruhe, neumann@iism.uni-karlsruhe.de

A MODEL OF PREFERENCE ELICITATION: THE CASE OF DISTRIBUTED RESOURCE ALLOCATION

Stößer, Jochen, Institute of Information Systems and Management, Universität Karlsruhe (TH), Englerstr. 14, 76131 Karlsruhe, Germany, jochen.stoesser@kit.edu

Neumann, Dirk, Chair for Information Systems Research, Albert-Ludwigs-Universität Freiburg, Kollegiengebäude II, Platz der Alten Synagoge, 79085 Freiburg, Germany, dirk.neumann@is.uni-freiburg.de

Abstract

Market mechanisms are deemed promising for distributed resource allocation settings by explicitly involving users into the allocation process. The market considers the users' and providers' valuations to generate efficient resource allocations and prices. In theory, valuations are assumed to be known to the user. In practice, however, this is not the case. It is a complex burden for both users and providers to assess their true valuation for a certain combination of resources and services and to efficiently communicate this valuation to the market.

This paper contributes to the theory of designing distributed allocation models in that (i) we propose a model for preference elicitation, which allows users and providers to assess their valuations as a function of their resource requirements and strategic considerations, (ii) we show how this model can be encoded within so-called bidding agents which interact with the market on behalf of the user, and (iii) we evaluate our approach in a numerical experiment to illustrate how the bidding agent adapts to the dynamic market situation. As this evaluation shows, the model outperforms technical schedulers and can thus be used for decision support in electronic markets.

Keywords: Distributed Resource Allocation, Preference Elicitation, Automated Trading.

1 INTRODUCTION

In recent times, enterprises have been facing increased pressure to host resource-demanding applications that exhibit fluctuating utilization patterns. At the same time they are forced to provide flexible infrastructures at low cost. A meta-study by IT critic Nicholas Carr (2005) revealed that enterprises only utilize on average 40-50% of their data centres' storage capacity and only 10-35% of their available processing power.

Grid and cluster technologies help enterprises deal with this dilemma. Enterprises no longer need to build up vast server farms in order to be able to serve few peak loads on their system. Instead, they only need to accommodate the basic load and tap into external computing, storage and application services on demand. The possible cost savings have been estimated to amount to 30% of total IT spending (Minoli 2004). Such grids can be located within enterprises to serve the various business units and departments, or spread across organizational boundaries to allow for dynamic resource (re-) allocations across enterprises. Prominent examples for the industry take-up of grid technology are the initiatives of Sun Microsystems, Google, IBM and Amazon. With its network.com platform, Sun offers computing resources for a fixed price of $1 per CPU hour. With Amazon's Elastic Compute Cloud (http://aws.amazon.com/ec2) and its Simple Storage Service (http://aws.amazon.com/s3), users can configure virtual servers and remotely run their applications and store their data on Amazon's infrastructure, only paying for what they use.

However, so far only few customers have adopted these offers. While Sun and Amazon both employ fixed pricing schemes, there is doubt whether this static model can adequately deal with the dynamic nature, which is inherent to grids as demand and supply are highly fluctuating. With fixed prices, users cannot express their priorities (i.e. valuations) for resources, thus leading to inefficient allocations and unrealized profit (Lai 2005).

Market mechanisms are deemed promising to provide a better fit to this dynamic setting by explicitly involving users into the allocation process. The market considers the users' and providers' valuations in order to generate efficient resource allocations and prices. Consequently, the market takes the responsibility of producing prices, which reflect the true scarcity of resources by adequately matching demand and supply. While market mechanisms exhibit compelling features, two important building blocks are missing which hampers their practical use: *preference elicitation* and *automated trading*. It is a complex burden for both users and providers to (i) assess their true valuation for a certain combination of resources and services and to (ii) efficiently communicate this valuation to the market.

While we focus on grid settings, our results are fundamental for all market-based business relationships, where users have to formulate bids. Literature frequently assumes that the preferences are fully known to the user. We will relax this strong assumption by proposing a model for preference elicitation. As such, this paper contributes to the theory of designing distributed allocation models in that

- We propose a model for preference elicitation, which allows users and providers to assess their valuations as a function of their resource requirements and strategic considerations.
- We show how this model can be encoded within so-called bidding agents, which interact with the market on behalf of the user.
- We evaluate our approach in a numerical experiment to illustrate how the bidding agent adapts to the dynamic market situation and how even fuzzy approaches can lead to more efficient allocations than technical schedulers.

The paper is structured as follows. In Section 2, we briefly discuss previous work on preference elicitation and automated bidding. Subsequently, we introduce the tasks of a bidding agent and existing approaches to preference elicitation from the marketing research domain in Section 3. In Section 4, we present our model which illustrates one approach for how such bidding agents can combine preference elicitation and trading protocols to dynamically adapt to the situation on the market. Clearly, this model represents just one approach; others are also possible. Thus, we need to thoroughly evaluate our approach via numerical experiments. This evaluation is covered in Section 5, where we present a sample market mechanism and

show the interdependency between the mechanism, the competition in this market, and the bidding agent's behaviour. Section 6 summarizes the paper and points to future research directions.

2 RELATED WORK

As pointed out in the introduction, we combine two aspects of the interaction between human users and the market for resources and services: *preference elicitation* and *automated trading*. Probably the largest body on preference elicitation stems from the domain of combinatorial auctions (cf. Conen and Sandholm 2001, Zinkevich et al. 2003, Parkes 2005, Nisan and Segal 2005). However, this previous work focuses on a separate issue: If users know their valuation, but communication between the users and the market is costly, how to efficiently query users for their valuations given the specific structure of the underlying allocation problem. This problem is inherent to combinatorial auctions, where the user has to submit valuations for all 2^{n-1} possible bundles which can be composed of n goods. Different from this literature, we use the term preference elicitation to denote the users' problem of determining their *true* valuation, i.e. questions such as "What am I willing to pay for using a server with application X, a dual-core processor and 2 GB of memory for one hour?". There is currently not much research available in this area, which is surprising as it is a prerequisite for any market-based approach.

Preference elicitation has been widely studied in the area of decision / negotiation support systems, where participants have to decide about their preference for an alternative or a good that is specified by multiple attributes. Typically, Multi-Attribute Utility Theory (MAUT) is used to aggregate the single attributes' values to an overall score or ranking (e.g. Bichler and Kaukal 1999). In MAUT, the negotiator's utility function is modelled as the weighted sum over the attributes' utility levels. In the INSPIRE (Kersten and Noronha 1999) and NEGOISST (Jertila and Schoop 2005) negotiation systems, a so-called hybrid conjoint analysis is applied (Green and Krieger 1996) to determine the attributes' relative importance. We defer a detailed discussion of conjoint analysis to Section 3.1. Vetschera (2007) investigates whether the preference structure of negotiators has an impact on their behaviour and the ultimate outcome.

Similar to our approach, Byde et al. (2003) also try to combine the problems of determining the value for specific resources and of allocating these scarce resources via markets in an automated manner so as to maximize the overall system value. In their setting, a utility centre operator has signed (long-term) service level agreements with various users and now has to decide how to serve these requests based on its actual situation. Byde et al. (2003) do not introduce bidding agents on the users' side, but solely on the provider's side. Each service request is represented by an application agent and a business agent. The application agent is responsible for monitoring and predicting the applications resource consumption. The business agent then translates this prediction into an estimated business value (from the provider's perspective) based on the underlying service level agreement, agreed metrics, penalties, rewards etc.

MacKie-Mason and Wellman (2006) study the automation of the user-market interaction by means of trading agents. By equipping users with such (at least partially) automated tools, the communication with the market can be drastically simplified since human users do not constantly need to monitor the market outcome and update their requests. One prominent outcome of this research is the TAC trading agent competition (www.sics.se/tac/) where research teams compete in designing trading agents for a specific market mechanism.

In summary and to our best knowledge, hitherto there has been no research that tries to combine the issues of preference elicitation and automated trading from the user's perspective in an auction context.

3 PREFERENCE ELICITATION

We propose the use of bidding agents that essentially have three levels of "intelligence":

1. **Prediction of resource demand:** The bidding agent learns the technical requirements of the grid application. This information could either be *static*, e.g. the user could manually configure the agent with the required amount of processing power, memory, and storage. But the agent could also *dynamically adapt*

its information, e.g. by estimating the resource requirements based on historical information using statistical methods (Degermark et al. 1997, Smith et al. 1998).

2. **Preference elicitation:** The bidding agent estimates the user's valuation for a specific application and combination of resources. This will be the core of this paper.

3. **Automated bidding:** The bidding agent interacts with the market in an automated fashion in order to obtain the right resource level – possibly according to a workflow of tasks – and considering the current market situation to achieve the best possible price.

The agent can then use information about the market outcome and the application's resource consumption to refine its estimates. For example, after the market has cleared, the agents can refine their bidding strategies. Furthermore, the user could review the bidding agent's preference elicitation by rating its accuracy, and the agent can compare the applications resource consumption with its prediction.

While we will focus on bidding agents for users in the remainder, note that the same principles can be applied to design bidding agents for the provisioning side of the market. We model our setting as a multi-attribute combinatorial allocation problem. A grid application is typically characterized by multiple attributes, such as the required amount of CPU, memory, storage, bandwidth, software libraries etc. The application can only be run if all such required resources are obtained in the right quality (Schnizler et al. 2008, Subramoniam et al. 2002).

More formally, we assume that an application A can be characterized by a finite set of attributes $X^A = \left(X_1^A, X_2^A, \dots, X_n^A\right) \in \mathbb{N}^n = \Omega$, where Ω is the space of possible application profiles.

Multi-attribute valuation function: Let application A be characterized by the finite set X^A of attributes. Then the user's valuation v^A of user i for this application A is a function $v^A: \Omega \longrightarrow \mathbb{R}$ with

$$v^A(x)\begin{cases} > 0, & if\ x_j \geq x_j^A, j = 1, \dots, n \\ = 0, & else \end{cases}$$

In the following, we will assume that the bidding agent has some estimate of application A's resource profile x^A, and we will focus on the preference elicitation and the automated bidding as well as the iterative refinement of the estimated user valuations.

The bidding agent essentially faces two decision or learning problems:

- What is user i's valuation function?
- How should the agent interact with the market to obtain the resource set x^A so as to maximize its user's benefit?

In this section, we will discuss several preference elicitation approaches which can be used to estimate a user's valuation for a specific application and a combination of attributes.

3.1 Conjoint Analysis

Conjoint analysis has its roots in the 1960's and 70's and has become the most widely used technique in marketing research to measure consumer preferences (Luce and Tukey 1964, Green and Rao 1971, Marder 1999). Conjoint analysis allows to ask the user "What if" questions. The technique begins by defining the number of attributes and possible values, so-called "attribute levels". The aim of the conjoint analysis is to estimate the user's value for each attribute. The analysis essentially creates a number of attribute combinations and attribute levels, so-called "profiles" (e.g. specific products), and asks the user to evaluate these profiles, e.g. by ranking or rating the profiles. For instance, a computing server may have the attributes CPU, memory, and bandwidth. Conjoint analysis would then generate specific server profiles, e.g. (price/hour, CPU, memory, bandwidth) = ($1, 2, 2 GB, 10 MB/s), ($2.5, 4, 3 GB, 100 MB/s) etc.

The key to conjoint analysis is the "partitioning assumption" (Marder 1999), which refers to the aggregation of the values for the attribute levels makes up the user's value for a specific profile. To simplify the analysis, one possibility is to follow the approach by Keeney and Raiffa (1976) and to use

additive independent valuation functions, where each resource (attribute) represents an additive term multiplied by a scale factor which encodes the relative importance of this resource attribute in comparison to the other resource attributes:

Additive independent valuation function: The attributes $X_1^A, \dots, X_n^A$ are mutually independent and v^A can be written as

$$v^A(x) = \begin{cases} \alpha_1^A x_1 + \alpha_2^A x_2 + \cdots + \alpha_n^A x_n, & if\ x_j \geq x_j^A, j = 1, \dots, n \\ 0, & else \end{cases}$$

with scale factors $\alpha_j^A \in \mathbb{R}, j = 1, \dots, n$.

Conjoint analysis estimates the user's value for a certain attribute level by performing regression analyses on the user's feedback to the presented attribute profiles.

The main drawback of this method is that the full-profile method presents all possible profiles to the user, which is clearly infeasible already for small numbers of attributes and attribute levels. Thus, as pointed out above, especially negotiation support systems employ hybrid and adaptive models, which restrict the number of profiles and mathematically determine the best profiles to present to the user (Green and Krieger 1996). Moreover, conjoint analysis relies on the partitioning assumption to be able to link the user's evaluation of the profile to the valuations for the individual attributes. But it is problematic to assume a common relationship between the overall profile and the attributes across all users.

3.2 Analytical Hierarchy Process

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 (Saaty 1980). The user has to compare the relative importance of attributes in a pairwise manner based on a predefined scale from 1, 3, …, to 9 (and $1, \frac{1}{3}, \dots, \frac{1}{9}$ respectively), e.g. "CPU is 3 times as important as memory". Ultimately, a matrix is generated containing all pairwise comparisons, from which a vector with the various attributes' relative importance is computed, e.g. (CPU, memory, bandwidth) = $\left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right)$. Saaty has also proposed a method to check the consistency of the user's comparisons based on the matrix's eigenvalue.

The analytical hierarchy process suffers from the large number of pairwise comparisons that the user has to perform to generate the matrix from which the relative weights are computed. With n attributes, the user essentially has to do $n - 1$ comparisons. Furthermore, while the method ultimately gives the relative weights of the attributes, the question remains about the *value* of a specific combination of attributes.

In summary, both conjoint analyses and the analytical hierarchy process are not appropriate for practical auction-based settings with a large number of attributes and attribute values and frequent trades. Furthermore, they depend on the assumption about a specific structure of the user's valuation function, e.g. an additive independent valuation function, which seems somewhat arbitrary.

4 THE MODEL

Thus, we propose to take an evolutionary approach. This approach is based on the assumption that a consumer who wants to buy some good does generally not exactly know his valuation for this good, i.e. if asked, the user would not be able to state a valuation. Instead, the user only has a rough estimate of his own true valuation. That is, when the consumer is confronted with an offer and a specific price, he decides whether to accept the given price, or to continue his search and look for alternative offers.

We adopt a similar approach in that the user initially indicates the request's priority, e.g. "high", "medium" or "low". After each market clearing, the agent presents the results to the user, and the user may indicate whether he is satisfied with the outcome or not. Based on this user feedback, the agent iteratively updates

its estimate of the user's valuation. So step by step, the agent approximates the results of a full-profile conjoint analysis and successively refines its estimates.

The evolutionary preference elicitation process goes as follows:

1. **Initialization:** In contrast to presenting all possible profiles to the user a priori, the agent initially assumes a valuation v_0^A for application A based on this application's resource specification x^A, the current price for each resource which is published by the market mechanism in a price vector $p_0 \in \mathbb{R}_+^n$, and an initial priority $\theta_0^A \in \mathbb{N}$ set by the user: $v_0^A(\theta_0^A, p_0, x^A) = \theta_0^A p_o^T x^A$.

If the price is only published for a bundle of resources, we assume that the market imputes and publishes prices for the individual resources (cf. Xia et al. 2004).

For example, a given application A requires 2 CPUs and 1 GB of memory for one minute, i.e. $X^A = (CPU, Memory, Runtime\ in\ mins)$ and $x^A = (2,1,1)$. CPU has last been traded for \$1 per hour, consuming 1 GB of memory for one hour cost \$2. The user indicates a medium priority $\theta_0^A = 2$. The bidding agent then initiates its estimate of the user's valuation for this task as $v_0^A(\theta_0^A, p_0, x^A) = 2 \cdot \left(\frac{\$1}{60} \cdot 2 + \frac{\$2}{60} \cdot 1\right) \approx \$0.133$.

This naïve approach might be extended by applying case-based reasoning (Hu and Haddawy 1998) or neural networks (Haddawy et al. 2003).

Subsequently, for each run j of the application:

2. **Bidding:** The agent bids on the market according to its estimate $v_j^A(\theta_j^A, p_j, x^A)$ and some bidding strategy.

3. **Refinement:** There are two possible outcomes of the bidding process:

a. *Successful*: The agent obtains the necessary resources at an overall price $p_{j+1}^T x^A = \sum_{i=1}^n p_{j+1,i} x_i^A \leq v_j^A$, and reports this information to the user. The user then indicates whether he is satisfied with the outcome or not, and the agent refines its estimate.

If the user indicates that he was satisfied with this price ($\theta_{j+1}^A = 1$), the agent does not update its estimate. If the user indicates that the price was too high ($\theta_{j+1}^A = 0$), the agent updates its estimate of v^A according to some update factor $\rho \in [0,1]$:

$$v_{j+1}^A(\theta_{j+1}^A, p_{j+1}, x^A) = \begin{cases} v_j^A(\theta_j^A, p_j, x^A), & if\ \theta_{j+1}^A = 1 \\ \rho p_{j+1}^T x^A, & if\ \theta_{j+1}^A = 0 \end{cases}$$

b. *Unsuccessful*: The agent was not able to obtain the necessary resources at a price $p_{j+1}^T x^A = \sum_{i=1}^n p_{j+1,i} x_i^A \leq v_j^A$.

If the user indicates that the price was indeed too high ($\theta_{j+1}^A = 1$), the agent does not update its estimate. If the user indicates that he would have preferred to pay that price rather than not getting the resources ($\theta_{j+1}^A = 0$), the agent updates its estimate of v^A with the current price:

$$v_{j+1}^A(\theta_{j+1}^A, p_{j+1}, x^A) = \begin{cases} v_j^A(\theta_j^A, p_j, x^A), & if\ \theta_{j+1}^A = 1 \\ p_{j+1}^T x^A, & if\ \theta_{j+1}^A = 0 \end{cases}$$

In our example, assume the agent was able to obtain the resources at a (total) price of $p_1 = \$0.08 < \$0.133 = v_0^A$ and the user indicated that this price was too high (Case a and $\theta_1^A = 0$). Then the agent updates its estimate to $v_1^A(\theta_1^A, p_1, x^A) = 0.9 \cdot \$0.08 = \$0.072$ for an update factor of $\rho = 0.9$.

The agent will iteratively try to converge its estimate to the user's true valuation and at the same time *assists* the user in identifying that valuation by presenting the user a series of simpler decision problems rather than forcing the user to directly reveal his valuation.

It is important to note that this approach does not depend on any assumption about the structure of the user's valuation function. But similar to conjoint analysis and the analytical hierarchy process, one critique may be that the user still needs to give feedback to the agent. But obviously such feedback loops cannot be avoided if the agent is to at least approximate the user's true valuation. However, we hypothesize that the agent can already obtain a good estimate with just a few such feedback loops. It will be an interesting question for future research if this assumption is valid, and if so, how many feedback loops are required to obtain a "good" estimate.

5 EVALUATION

As pointed out above, we hypothesize that grid users will generally not be able to state an exact number if asked for their valuation for a certain application. Instead, users will only approximate this valuation over time with the bidding agent's assistance. This inherently precludes a "hard" evaluation of the proposed evolutionary approach to preference elicitation, as we cannot simulate this process. However, we will evaluate two complementary aspects of grid markets:

- We will show that the use of grid markets leads to efficiency gains over simple technical schedulers, even if the bidding agent uses a simple rule for initializing its estimate about the user's valuation as presented above.
- We illustrate for a sample setting how the bidding agent strategically misreports the user's (estimated) valuation so as to maximize its user's benefit.

Of course the market outcome and the agent's behavior depend on the actual market mechanism. Thus, before we turn to our analysis, we will briefly introduce a sample mechanism.

5.1 The Market Mechanism

The following mechanism has been proposed by Stößer et al. (2007) and is tailored towards usage in grids (Amar et al. 2007). Market mechanisms essentially consist of three elements: the bidding language which determines how users and providers specify their resource requests and offers, the allocation scheme which assigns resource requests to offers, and the pricing scheme which determines corresponding payments.

5.1.1 The Bidding Language

A user j who would like to submit a computational job to the grid system reports the job's characteristics $(v_j, \underline{c}_j, \underline{m}_j, s_j, e_j)$ to the market mechanism where $v_j \in \mathbb{R}_+$ denotes j's valuation (i.e. j's maximum willingness to pay) per unit of computing power and time slot, $\underline{c}_j \in \mathbb{N}$ and $\underline{m}_j \in \mathbb{N}$ the minimum required amount of computing power and memory respectively, and $s_j \in \mathbb{N}$ and $e_j \in \mathbb{N}$ specify the job's estimated runtime (start and end). We require the market mechanism to make atomic allocations in the sense that each job can only be executed if there are sufficient resources available in all requested time slots. Furthermore, jobs can potentially be migrated between several compute nodes over time but each job can only be executed on one node at a time.

A provider n who would like to contribute a compute node to the grid system reports the node's characteristics $(r_n, \bar{c}_n, \bar{m}_n, \varepsilon_n, \lambda_n)$ to the market mechanism where $r_n \in \mathbb{R}_+$ specifies this node's (pretended) reservation price per unit of computing power and time slot, $\bar{c}_n \in \mathbb{N}$ and $\bar{m}_n \in \mathbb{N}$ the maximum amount of computing power and memory available on this node, and $\varepsilon_n \in \mathbb{N}$ and $\lambda_n \in \mathbb{N}$ the time frame during which the node can be accessed. Given sufficient resources, we assume that each node is able to execute multiple jobs in parallel.

5.1.2 The Allocation Scheme

The objective of the market is to generate efficient resource allocations. This can be modelled as a combinatorial allocation problem as proposed in Stößer et al. (2007). However, due to the resulting computational complexity, Stößer et al. (2007) propose a greedy heuristic as follows:

1. The resource requests are sorted in non-ascending order of their valuations v_j. Resource offers are sorted in non-descending order of their reservation prices r_n.

2. The heuristic loops over the ranked list of requests and successively constructs a feasible allocation schedule by assigning the requests with the highest valuations to the cheapest offers.

5.1.3 The Pricing Scheme

Subsequently, the allocation schedule generated by this heuristic needs to be complemented by corresponding prices. K-Pricing was introduced in Schnizler et al. (2008). The basic idea is to distribute the welfare generated by the allocation algorithm between users and resource providers according to a factor $k \in [0,1]$. For instance, assume an allocation of resources from a specific provider to a specific user. The user values these resources at $\$10$ while the provider has a reservation price of $\$5$. Then the (local) welfare is $\$10 - \$5 = \$5$ and $k \cdot \$5$ of the surplus is allotted to the user – thus having to pay $\$10 - k \cdot \$5$ – and $(1-k) \cdot \$5$ is allotted to the provider – thus receiving $\$5 + (1-k) \cdot \$5$. Besides allowing for fairness considerations, the main advantage of K-Pricing is that it can be determined in polynomial runtime. On its downside, however, it only yields approximately truthful prices and payments on both sides of the market, i.e. users and providers may benefit from misreporting their valuations.

5.2 Data Generation

The problem with numerical evaluations of grid markets is that there is no log of real-world workloads which contains all parameters according to our bidding language specified above, in particular as regards the users' valuations. We thus created synthetic workloads using the distributions specified in Table 1, which is in line with Feitelson (2002).

Parameter	*Distribution*
Request start time $s_j$and runtime e_j	Binomial B(5, 0.5)
Request CPU requirement $\underline{c_j}$	Binomial B(5, 0.5)
Request memory requirement $\underline{m_j}$	Lognormal L(4, 0.15)
Request valuation v_j	Uniform U(1, 30)
Offer start time ε_n	Binomial B(4, 0.5)
Offer availability λ_n	Binomial B(8, 0.5)
Offer CPU $\overline{c_n}$	Binomial B(10, 0.5)
Offer memory $\overline{m_n}$	Lognormal L(5, 0.2)
Offer reservation price r_n	zero reservation prices

Table 1. Distributions used for generating the synthetic workloads.

We analyze four settings with varying degrees of competition: 20 requests and 20 offers, 40 requests and 20 offers, 60 requests and 20 offers, and 80 requests and 20 offers. For each of these four settings, we generated 200 problem instances. In the remainder, we will report the averages across these 200 runs to account for stochastic outliers.

5.3 Data Analysis

5.3.1 Efficiency Gains from Preference Elicitation

In this subsection, we analyze the performance of our preference elicitation scheme with respect to allocative efficiency, which is defined as the aggregated utility across all users and providers: How well does our preference elicitation scheme perform (i) compared to the market mechanism presented in

Subsection 5.1 if users know their valuations and truthfully submit these to the mechanism and (ii) compared to a technical scheduler which does not consider user preferences? As mentioned above, it is not possible to evaluate our iterative preference elicitation approach as such. Instead, we will evaluate the initialization step in which the agent estimates the user's valuation based on the application's resource requirements, current market prices, and a fuzzy priority assigned by the user.

We configure the bidding agent to initialize its estimate of the user's valuation as follows. Since the heuristic from Subsection 5.1 ranks resource requests according to their valuation *per CPU and timeslot*, the bidding agent initializes its estimate of the user j's true valuation as $v_{j0}(\theta_{j0}, p_0, x_j) = \theta_{j0} p_{0,CPU}$. We assume that users can distinguish between three job priorities $\{1,2,3\}$, e.g. "high", "medium" and "low" priority. The initial user priorities are (recall that we draw the true valuations from U(1, 30))

$$\theta_{j0} = \begin{cases} 1, if\ 1 \leq v_j \leq 10 \\ 2, if\ 11 \leq v_j \leq 20 \\ 3, if\ 21 \leq v_j \leq 30 \end{cases}$$

The effect is that, assuming all agents apply this initialization rule, the requests get clustered into three blocks, requests with $v_{j0}(\theta_{j0}, p_0, x_j) = p_{0,CPU}$, $v_{j0}(\theta_{j0}, p_0, x_j) = 2p_{0,CPU}$, and $v_{j0}(\theta_{j0}, p_0, x_j) = 3p_{0,CPU}$. For example, let job j require one CPU and 100 MB memory for three timeslots. The user's true valuation is \$3 per timeslot. One CPU has last been traded for \$1 per timeslot. Then the user indicates an initial priority of 1, and the agent estimates a valuation of $v_{j0}(\theta_{j0}, p_0, x_j) = 1 \cdot \$1 = \$1$. Ultimately, assuming truthful behaviour, the agent reports $(v_{j0}(\theta_{j0}, p_0, x_j), \underline{c}_j, \underline{m}_j, s_j, e_j)$ to the market.

Technical schedulers only aim at maximizing resource utilization or balancing the system load. As a proxy for such technical schedulers we implemented a scheduler which randomly assigns jobs to feasible nodes that offer sufficient resources in the right timeslots.

The numerical results with respect to allocative efficiency are reported in Table 2.

Number of requests and offers (ratio of allocated requests)	*Efficiency with true valuations*	*Efficiency with initialized preference elicitation*	*Efficiency of randomized (technical) scheduler*
20 requests, 20 offers (97.3%)	3,687.08	3,679.11 (99.8%)	3,664.66 (99.4%)
40 requests, 20 offers (82.4%)	6,852.75	6,693.97 (97.7%)	6,013.25 (87.7%)
60 requests, 20 offers (61.6%)	8,734.67	7,885.82 (90.3%)	6,561.02 (75.1%)
80 requests, 20 offers (48.8%)	9,776.49	8,366.91 (85.6%)	6,716.23 (68.7%)

Table 2. Efficiency generated by the various approaches.

We successively increased the competition in the market, indicated by the first column. With 20 requests and 20 offers, almost all requests get allocated. But with 80 requests and 20 offers, only 48.8% of the requests can be accommodated. The economic performance of technical schedulers will decrease as competition in the market increases. If competition is low, technical schedulers will be able to accommodate most of the requests, so the negative effect of not considering the users' valuations is comparably small. But obviously this negative effect increases with the scarcity of the resources. This general reasoning is confirmed by our results. For the most competitive setting with 80 requests and 20 offers, the technical scheduler only achieves 68.7% of the efficiency which the market mechanism achieves with true valuations. Interestingly, despite the simple initialization rule and without considering the iterative refinement of the agents' estimates, the market mechanism combined with bidding agents still achieves 85.6% of this benchmark. This result underlines the case for considering economic principles in the allocation of grid resources, even if the hard informational assumptions regarding user valuations are relaxed.

5.3.2 Strategic Considerations with Automated Bidding

In this subsection, we will illustrate the general idea for how bidding agents can implement strategic behaviour when interacting with the market in order to maximize the users' benefit. As outlined in Section 3, this is a learning task in itself which is complementary to preference elicitation. We assume the bidding

agent to have an estimate $v_{j0}(\theta_{j0}, p_0, x_j)$ about user j's true valuation. We model the one-dimensional action space of the agent to consist of a factor $\beta \in \mathbb{R}_+$. Consequently, the agent reports $(\beta \cdot v_{j0}(\theta_{j0}, p_0, x_j), \underline{c}_j, \underline{m}_j, s_j, e_j)$. The question is how the agent should choose β.

As is common in agent-based simulations, this choice can be modelled using reinforcement learning, e.g. Q-learning (Watkins 1989). This is basically a trial-and-error approach, where the agent tests different strategies (i.e. choices of β) during the *exploration phase*. After this exploration phase, the agent chooses the best strategy according to some policy, e.g. the β which resulted in the highest utility (the so-called epsilon-greedy selection policy). Then, during the *exploitation phase*, the agent plays this best strategy. In our setting, the agent may initially submit the true (estimated) valuation $v_{j0}(\theta_{j0}, p_0, x_j)$ to the market. Then it successively explores its strategy space by deviating from this true valuation, e.g. by only bidding 50% of its true estimate, 55%, 60%, up to 150%.

The results of this strategy ($\beta \in [0.5, 1.5]$) for our numerical evaluation are illustrated in Figure 1, where we again employed the heuristic above complemented by K-Pricing with $k = 0.5$. The graphs show the utility relative to truthful bidding for the four settings.

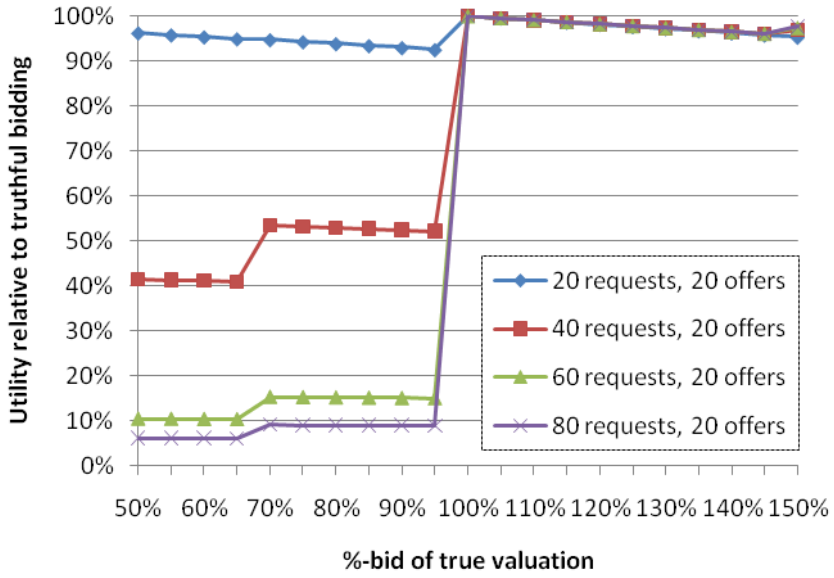

Figure 1. Efficiency gain from misreporting.

The results show that, for each setting, it is the best strategy on average to report the true estimate. Intuitively, the higher the competition, the higher the risk of not being allocated anymore if the agent reports a valuation below its true estimate, thus yielding a smaller utility for the user. The discrete priority values $\theta_{j0} \in \{1,2,3\}$ are reflected in the step-like shapes of the relative utility when underbidding ($0.5 \leq \beta < 1$). Assuming all agents apply the same initialization step with n discrete job priorities, we get n clusters of requests with identical $v_{j0}(\theta_{j0}, p_0, x_j)$ in the ranking of the allocation heuristic. Consequently, as soon as $\beta \cdot v_{j0}(\theta_{j0}, p_0, x_j)$ falls within the next lower cluster in the ranking, we get a drop in the relative utility.

If the agent overbids, there are two possible consequences. If the request was allocated with truthful bidding, then the request will still be allocated if the agent reports a higher valuation. However, with K-Pricing the user will receive a smaller fraction of the true welfare, and thus have a smaller utility. If the request was not allocated with truthful bidding, the request either remains outside of the allocation if the agent reports a higher valuation, thus leaving the user's utility unchanged, or the request gets allocated, yielding a positive utility as long as $\beta < 2$ with zero reservation prices (cf. K-Pricing rule). However, as the results show, the overall effect of overbidding is slightly negative on average. This property is important, as agents have no incentive to misrepresent their priority.

Overall, the results indicate that the agent should truthfully report $v_{j0}(\theta_{j0}, p_0, x_j)$ during the exploitation phase in our experimental setting.

6 CONCLUSION & OUTLOOK

The idea of applying market mechanisms to the allocation of scarce resources in dynamic and distributed settings is certainly not new. In such settings, markets exhibit many desirable features, in particular they promise to lead to more efficient allocations than traditional technical approaches. However, we argue that markets are facing two main obstacles:

- Users will generally not exactly know their true valuations for the resources, but only have a fuzzy estimate.
- Users cannot be expected to continuously monitor and interact with the market, but this interaction is costly. For instance, human users incur opportunity costs and have bounded perceptivity.

Surprisingly, these two barriers have largely been neglected in the vast body of literature on market mechanisms. Instead, existing work simply assumes perfect information on the users' side and costless interaction with the market. The aim of this paper was to take a first step towards overcoming this obvious gap between theory and real-world settings. We briefly discussed previous work on preference elicitation and automated bidding in Section 2. In Section 3, we introduced the learning tasks of bidding agents and existing approaches to preference elicitation from the field of marketing research. At the core of this paper, we proposed a new model for preference elicitation and automated trading in distributed settings. To evaluate the initialization step of this model and the strategic behaviour of bidding agents in more detail, we performed a numerical experiment with a sample market mechanism, whose results we reported and interpreted in the previous section. In summary, these results underline the case for considering market mechanisms in the allocation of distributed resources, even if the hard informational assumptions regarding user valuations are relaxed. Instead, we propose the use of bidding agents, which (i) assist the user in finding his true valuation using an evolutionary approach, and (ii) shield parts of the underlying infrastructure's and the market's complexity from the user by acting on his behalf.

This work suggests several natural extensions for future research. The preference elicitation task of the agent requires further analyses, in particular as regards the iterative refinement of the agent's estimates. How should the agent be parameterized, e.g. as regards the update parameter ρ and the discrete user priorities θ_j^A? As pointed out above, the use of case-based methods (Hu and Haddawy 1998) and neural networks (Haddawy et al. 2003) might be interesting approaches. And how many iterations does this process take to get within x% of the user's true valuation? Moreover, the automated trading of bidding agents offers interesting research questions. How does the agent behaviour depend on the specific market mechanism and vice versa? And what exploration and exploitation strategies should such agents apply? These questions can certainly not be answered with our rather static analysis, but call for agent-based simulations.

References

Amar, L., J. Stößer, A. Barak, D. Neumann (2007). Economically Enhanced MOSIX for Market-based Scheduling in Grid OS. Workshop on Economic Models and Algorithms for Grid Systems (EMAGS), 19 September, Austin, TX, USA.

Bichler, M. and Kaukal, M. (1999). Design and Implementation of a Brokerage Service for Electronic Procurement. Database and Expert Systems Applications (DEXA), 1-3 September, Florence, Italy, 618-622.

Byde, A., M. Sallé and C. Bartolini (2003). Market-Based Resource Allocation for Utility Data Centers. Technical Report HPL-2003-188, HP Laboratories Bristol, UK.

Carr, N. (2005). The End of Corporate Computing. Sloan Management Review, 46(3), 67-73.

Chen, L. and P. Pu (2004). Survey of Preference Elicitation Methods. Technical report IC/2004/67, Ecole Politechnique Federale de Lausanne, Switzerland.

Conen, W. and T. Sandholm (2001). Preference Elicitation in Combinatorial Auctions. ACM Conference on Electronic Commerce (EC'01), 14-17 October, Tampa, Florida, USA.
Degermark, M., T. Kohler, S. Pink and O. Schelen (1997). Advance Reservations for Predictive Service in the Internet. Journal on Multimedia Systems, 5(3), 177-186.
Feitelson, D.G. (2002). Workload Modeling for Performance Evaluation. In Performance Evaluation of Complex Systems: Techniques and Tools. LNCS, 2459, 114-141, Springer Verlag.
Green, P.E. and R. Rao (1971). Conjoint Measurement for Quantifying Judgmental Data. Journal of Marketing Research, 8, 355-363.
Green, P.E. and A.M. Krieger (1996). Individualized Hybrid Models for Conjoint Analysis. Management Science, 42(6), 850-867.
Ha, V. and Haddawy, P. (1998). Toward Case-Based Preference Elicitation: Similarity Measures on Preference Structures. 14th Conference on Uncertainty in Artificial Intelligence, 24-26 July, Madison, Wisconsin, USA.
Haddawy, P., Ha, V., Restificar, A., Geisler, B. and Miyamoto, J. (2003). Preference Elicitation via Theory Refinement. Journal of Machine Learning Research, 4, 317-337.
Jertila, A. and Schoop, M. (2005). Electronic Contracts in Negotiation Support Systems: Challenges, Design and Implementation. IEEE International Conference on E-Commerce Technology (CEC), 19-22 July, Munich, Germany, 396-399.
Keeney, R.L. and H. Raiffa (1976). Decisions with Multiple Objectives: Preferences and value tradeoffs. Cambridge University Press.
Kersten, G. and Noronha, S.J. (1999). WWW-Based Negotiation Support: Design, Implementation, and Use. Decision Support Systems, 25(2), 135-154.
Lai, K. (2005). Markets are Dead, Long Live Markets. ACM SIGecom Exchanges, 5(4), 1-10.
Luce, R.D. and J.W. Tukey (1964). Simultaneous Conjoint Measurement: A New Type of Fundamental Measurement. Journal of Mathematical Psychology, 1(1), 1-27.
MacKie-Mason, J. and M.P. Wellman (2006). Automated Markets and Trading Agents. In Handbook of Computational Economics, Volume 2: Agent-Based Modeling, North Holland.
Marder, E. (1999). The Assumptions of Choice Modelling: Conjoint Analysis and SUMM. Canadian Journal of Marketing Research, 18, 3-14.
Minoli, D. (2004). A Networking Approach to Grid Computing. John Wiley & Sons. Inc., New York.
Nisan, N. and I. Segal (2005). Exponential Communication Inefficiency of Demand Queries. 10th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2005), 10-12 June, Singapore.
Parkes, D. (2005). Auction Design with Costly Preference Elicitation. Annals of Mathematics and Artificial Intelligence, 44(3), 269-302.
Saaty, T. (1980). The Analytical Hierarchy Process. Mc Graw Hill, New York.
Schnizler, B., D. Neumann, D. Veit and C. Weinhardt (2008). Trading Grid Services – A Multi-Attribute Combinatorial Approach. European Journal of Operational Research, 187(3), 943-961.
Stößer, J., D. Neumann and C. Weinhardt (2007). A Truthful Heuristic for Efficient Scheduling in Network-Centric Grid OS. 15th European Conference on Information Systems (ECIS), 7-9 June, St. Gallen, Switzerland.
Smith, W., I. Foster. and V. Taylor (1998). Predicting Application Run Times Using Historical Information. IPPS/SPDP 1998 Workshop on Job Scheduling Strategies for Parallel Processing, Orlando, Florida, USA.
Subramoniam, K., M. Maheswaran and M. Toulouse (2002). Towards a Micro-Economic Model for Resource Allocation in Grid Computing Systems. Canadian Conference on Electrical and Computer Engineering, 12-15 May, Winnipeg, Canada.
Vetschera, R. (2007). Preference Structures and Negotiator Behavior in Electronic Negotiations. Decision Support Systems, 44(1), 135-146.
Watkins, C.J.C.H. (1989). Learning From Delayed Rewards. PhD Thesis, University of Cambridge.
Xia, M., G.J. Koehler and A.B. Whinston (2004). PricingCcombinatorial Auctions. European Journal of Operational Research, 154(1), 251-270.
Zinkevich, M., A. Blum and T. Sandholm (2003). On Polynomial-Time Preference Elicitation with Value Queries. ACM Conference on Electronic Commerce (EC'03), 9-12 June, San Diego, California, USA.