Agent-based decision support for actual-world procurement scenarios

Juan A. Rodríguez-Aguilar
IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Scientific Research Council
jar@iiia.csic.es

Andrea Giovanucci, Antonio Reyes-Moro,
Francesc X. Noria, Jesús Cerquides
ISOCO, S. A.
{andrea,toni,fxn,cerquide}@isoco.com

Abstract

Multi-item, multi-unit negotiations in industrial procurement pose serious challenges to buying agents when trying to determine the best set of providering agents' offers. Typically, a buying agent's decision involves a large variety of constraints that may involve attributes of separate items as well as attributes of different, multiple items. In this paper we present Ibundler*, an agent-based service offered to buying agents to help them determine the optimal bundle of received offers based on their constraints and preferences. In this way, buying agents are relieved with the burden of solving too hard a problem (the problem can be proven to be NP) and concentrate on strategic issues.* IBundler *is intended as a negotiation service for buying agents and as a winner determination service for reverse combinatorial auctions with side constraints. To the best of our knowledge this is the first agent service for such type of complex negotiation scenarios. In addition to this, the* Ibundler *agent was the winner of the Best Application of the 2003 Agentcities Technology Competition[18].*

1. Introduction

With the advent of the Internet, agent researchers envisioned a promising future to software agents in a large variety of e-commerce settings. Business automation, decision-making and enterprise integration have been widely claimed, among others, as suitable tasks for agents. Despite the hype, it is our view that indeed there are e-commerce scenarios where agent technology can prove valuable. In particular, agent technology can contribute to the automation of complex tasks and to the assistance of parties involved in intricate decision-making processes in procurement scenarios. One particular, key procurement activity carried out by most companies concerns the negotiation of both direct and indirect goods and services.

Although negotiation is a key procurement mechanism, to the best of our knowledge most agent-based services deployed have focused on infrastructure issues related to negotiation protocols and ontologies. Thus, the lack of agent-based decision support for trading agents that help improve current trading practices hinders the adoption of agent technology in procurement scenarios.

Consider the problem faced by a buying agent when negotiating with providing agents. In a negotiation event involving multiple, highly customisable goods, buying agents need to express relations and constraints between attributes of different items. On the other hand, it is common practice to buy different quantities of the very same product from different providing agents, either for safety reasons or because offer aggregation is needed to cope with high-volume demands. This introduces the need to express business constraints on the number of supplying agents and the amount of business assigned to each of them. Not forgetting the provider side, supplying agents may also impose constraints or conditions over their offers. Offers may be only valid if certain configurable attributes (f.i. quantity bought, delivery days) fall within some minimum/maximum values, and assembly or packing constraints need to be considered. Once the buying agent collects offers, he is faced with the burden of determining the winning offers. The problem is essentially an extension of the combinatorial auction (CA) problem, which can be proven to be NP [3]. It would be desirable to relieve buying agents from solving such a problem.

In this paper we have tried to make headway in this direction by deploying an agent-based decision support service acting as a combinatorial negotiation solver (solving the winner determination problem) for both multi-item, multi-unit negotiations and auctions. Thus, the service can be employed by both buying agents and auctioneers in combinatorial negotiations and combinatorial reverse auctions [19] respectively. Furthermore, it extends current combinatorial auction models by accommodating both operational constraints and attribute-value constraints. At this aim, new ontological issues needed to be considered in order to empower the expressiveness offered by negotiation objects and offers to incorporate buyers' and providers' business constraints. Therefore, our approach required the extension of state-of-the-art ontologies for automated negotiation [13].

To the best of our knowledge, iBundler represents the first agent-based service for multi-item, multi-unit negotiations. Its innovative deployment over the Agentcities [18] framework has been recognised with a prize as the winner of the Best Application of the 2003 Agentcities Technology Competition.

The paper is organised as follows. Section 2 introduces the market scenario where buyers and traders are to negotiate, along with the requirements and constraints they may need. Next, the actual computational realisation of agent-based decision support for combinatorial reverse auctions is thoroughly detailed in Section 3, and conveniently illustrated via an example. Section 4 provides some hints on the expected performance of the service. Finally, Section 5 summarises our contributions and future work.

2. Market characterisation

Although the application of combinatorial auctions (CA) to e-procurement scenarios (particularly reverse auctions) may be thought as straightforward, the fact is that there are multiple new elements that need to be taken into consideration. These are new requirements explained by the nature of the process itself. While in direct auctions, the items that are going to be sold are physically concrete (they do not allow configuration), in a negotiation event involving highly customisable goods, buying agents need to express relations and constraints between attributes of different items. On the other hand, it is common practice to buy different quantities of the very same product from different supplying agents, either for safety reasons or because offer aggregation is needed to cope with high-volume demands. This introduces the need to express constraints on the number of supplying agents and the amount of business assigned to each of them. Not forgetting the provider side, supplying agents may also impose constraints or conditions over their bids/offers. Offers may be only valid if certain configurable attributes (f.i. quantity bought, delivery days) fall within some minimum/maximum values, and assembly or packing constraints need to be considered.

Current CA reviewed do not model these features with the exception of [12][14], where coordination and procurement constraints can be modelled. The rest of work focuses more on computational issues (CA is an NP-hard problem [3]) than in practical applications to e-procurement. Suppose that we are willing to buy 200 chairs (any colour/model is fine) for the opening of a new restaurant, and at that aim we employ an e-procurement solution that launches a reverse auction. If we employ a state of the art CA solver, a possible resolution might be to buy 199 chairs from provider A and 1 chair from provider B, simply because it is 0.1% cheaper and it was not possible to specify that in case of buying from more than one supplier a minimum of 20 chairs purchase is required. On the other hand the optimum solution might tell us to buy 50 blue chairs from provider A and 50 pink chairs from provider B. Why, because although we had no preference over the chair's colour, we could not specify that regarding the colour chosen all chairs must be of the same colour. Although simple, this example shows that without means of modeling these natural constraints, solutions obtained are seen as mathematically optimal, but unrealistic and with a lack of common sense, thus obscuring the power of decision support tools, and preventing the adoption of these technologies in actual-world settings.

Next we detail the capabilities required by buyers in the kind of negotiation scenario outlined above. The requirements below are intended to capture buying agents' constraints and preferences and outline a powerful bidding language for providing agents:

1) *Negotiate over multiple items*. A negotiation event is usually started with the preparation of a request for proposal (RFQ) form. The RFQ form describes in detail the requirements (including attribute-values such as volume, quality specifications, dates as well as drawings and technical documentation) for the list of items (goods or services) defined by the negotiation event.

2) *Offer aggregation*. A specific item of the RFQ can be acquired from several providers simultaneously, either because not a single provider can provide with the requested quantity at requested conditions or because buyer's explicit constraints (see below).

3) *Business sharing constraints*. Buyers might be interested to restrict the number of providers that will finally trade for a specific item of the RFQ, either for security or strategical reasons. It is also of usual practice to define the minimum amount of business that a provider may gain per item.

4) *Constraints over single items*. Every single item within an RFQ is described by a list of negotiable attributes. Since: a) there exists a degree of flexibility in specifying each of these attributes (i.e. several values are acceptable) and b) multiple offers referring the very same item can be finally accepted; buyers need to impose constraints over attribute values. An example of this can be the following: suppose that the deadline for the reception of certain item A is two weeks time. However, although items may arrive any day within two weeks, once the first units arrive, the rest of units might be required to arrive in no more than 2 days after.

5) *Constraints over multiple items*. In daily industrial procurement, it is common that accepting certain configuration for one item affects the configuration of a different item, for example, when dealing with product compatibilities. Also, buyers need to express constraints and relationship between attributes of different items of the RFQ.

6) *Specification of providers' capacities*. Buyers cannot risk to award contracts to providers whose production/servicing capabilities prevent them to deliver overcommitted offers. At this aim, they must require to have providers' capacities per item declared.

Analogously, next we detail the expressivenes of the bidding language required by providers. The features of the language below are intended to capture providing agents' constraints and preferences.

7) *Multiple bids over each item*. Providers might be interested in offering alternate conditions/configurations for a same good, i.e., offering alternatives for a same request. A common situation is to offer volume-based discounts. This means that a provider submits several offers and each offer only applies for a minimum (maximum) number of units.

8) *Combinatorial offers*. Economy efficiency is enhanced if providers are allowed to offer (bid on) combination of goods. They might lower the price, or improve service assets if they achieve to get more business.

9) *Multi-unit offering*. Each provider needs to specify that they will only participate in trading if a minimum (maximum) amount of business is assigned to him.

10) *Homogeneous combinatorial offers*. Combinatorial offering may produce inefficiencies when combined with multi-unit offering. Thus a provider may wind up with an award of a small number of units for a certain item, and a large number of units for a different item, being both part of the very same offer (e.g. 10 chairs and 200 tables). It is desirable for providers to be able to specify homogeneity with respect to the number of units for complementary items.

11) *Packing constraints*. Packing units are also a constraint, in the sense that it is not possible to serve an arbitrary number of units (e.g. a provider cannot sell 27 units to a buyer because.his items come in 25-unit packages). Thus providers require to be capable of specifying the size of packing units.

12) *Complementary and exclusive offers*. Providers usually submit XOR bids, i.e., exclusive offers that cannot be simultaneously accepted. Also, there may exist the need that an offer is selected only if another offer is also selected. We refer to this situation as an AND bid. This type of bids allows to express volume-based discounts. For example, when pricing is expressed as a combination of base price and volume-based price (e.g. first 1000 units at €2.5 p.u. and then €2 each).

Obviously, many more constraints regarding pricing and quantity can be considered here. But we believe these faithfully address the nature of the problem. Actually, iBundler has been applied to scenarios where some of these constraints do not apply while additional constraints needed to be considered. This was the case of a virtual shopping assistant, an agent that was able to aggregate several on-line supermarkets and optimize the shopping basket. To do so, it was necessary to model the fact that delivery cost depends on the amount of money spent at each supermarket.

3. Agent service

This section details the realisation of the agent service as an agency. Firstly, we present the implementation of the winner determination problem for CA with side constraints as the core of the service. Secondly, we describe the architecture of the iBundler agency, along with a description of the protocols and the ontology employed by its agents to offer the negotiation service.

3.1. Winner determination

Consider the problem faced by a buying agent aiming at choosing the optimal set of offers sent over by supplying agents taking into account the features of the negotiation scenario described in Section 2. The problem is essentially an extension of the combinatorial auction (CA) problem in the sense that it implements a larger number of constraints and supports richer bidding models. The CA problem is known to be NP-complete, and consequently solving methods are of crucial importance. Many of the works reviewed in the literature adopt global optimal algorithms as a solution to the CA because of the drawbacks pointed out for incomplete methods. Basically two approaches have been followed: traditional Operations Research (OR) algorithms and new problem specific algorithms [1][4]. It is always an interesting exercise to study the nature of the problem in order to develop problem specific algorithms that exploit problem features to achieve effective search reduction. However, the fact is that the CA problem is an instance of the multi-dimensional knapsack problem MDKP (as indicated in [10]), a mixed integer program well studied by the operation research literature. It is not surprising that, as reported in [11], many of the main features of these problem specific new algorithms are rediscoveries of traditional methods in the operations research community. In fact, our formulation of the problem can be regarded as similar to the binary multi-unit combinatorial reverse auction winner determination problem in [19] with side constraints [14]. Besides, expressing the problem as a mixed integer programming problem with side constraints enables its resolution by standard algorithms and commercially available, thoroughly debugged and optimised software which have shown to perform satisfactorily for large instances of the CA problem.

With these considerations in mind, the core of our agent service has been modelled and implemented as a mixed integer programming problem. We have implemented two versions: a version using ILOG CPLEX 7.1 in combination with SOLVER 5.2; and another version using using iSOCO's Java MIP modeller that integrates the GLPK library [15]. In both cases it takes the shape of a software component. Hereafter we shall refer to this component as the iBundler *solver*.

3.2. Architecture

The iBundler service has been implemented as an agency composed of agents and software components that cooperatively interact to offer a decision-support service for highly-constrained negotiation scenarios. iBundler can act as a combinatorial negotiation solver for both multi-item, multi-unit negotiations and auctions. Thus, the service can be employed by both negotiating agents and auctioneers in combinatorial auctions.

Figure 1 depicts the components of the agency, along with the fundamental interactions of buying and providing agents with the service[1]. Next we make explicit the main functionality of its members:

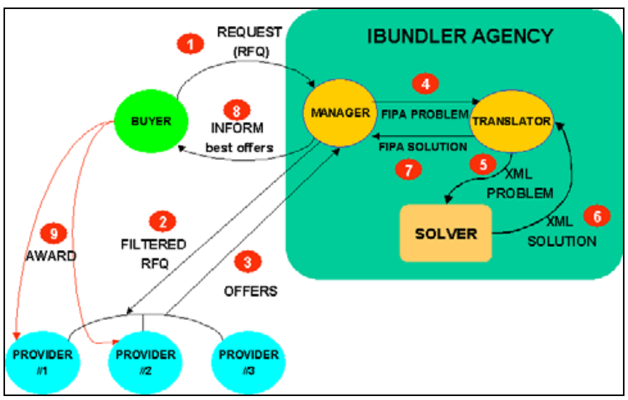

Figure 1. Architecture of the iBundler agency

Manager agent. Agent devoted to providing the solution of the problem of choosing the set of bids that best matches a user's requirements. There exists a single *Manager* agent per user (buying agent or auctioneer) offering them the following services: brokering service to forward buying agents' requirements (RFQs) to selected providing agents capable of fulfilling them; collection of bids; winner determination in a combinatorial negotiation/auction; award of contracts on behalf of buying agents. Furthermore, the manager agent is also responsible for: bundling each RFQ and its bids into a negotiation problem in FIPA-compliant [20] format to be conveyed to the *Translator* agent; and to extract the solution to the negotiation problem handled back by the *Translator* agent.

Translator agent. It creates an XML document representing the negotiation problem in a format understandable by the *Solver* departing from the FIPA-compliant description received from the *Manage*r. It also translates the solution returned by the *Solver* into an object of the ontology employed by user agents. It is the bridge between the language spoken by user agents and the language spoken by *Solver*.

Solver component. It offers an XML language for expressing offers, constraints, and requirements. The XML specification is parsed into an MIP formulation and solved using available MIP solvers as described in Section 4.1.

Our design manages to separate concerns among the three members of the agency. On the one hand, the *Manager* is strictly devoted to coordination. It represents the façade of the service. Besides, since every negotiation requested by a buying agent makes the agency create an instance of the Manager, the service can cope with scalability issues. Thus, if the service is heavily accessed, *Managers* can synchronise to queue tasks for the *Translator*. This is in charge of relieving both Managers and *Solver* from the burden of translating FIPA-compliant specifications into the XML language required by *Solver*. We pursued to have Solver exclusively dedicated to handle computationally expensive negotiation problems as it is. Notice too that *Solver* has been implemented as a software component because it was intended to serve for two purposes: as the core component of the iBundler agency, and as the winner determination component in a commercial sourcing application [2].

Finally, in order to implement the iBundler service we used the following technologies:

- We used JADE as the software tool to implement agents, and at the same time as the platform where the agency resides.
- We employed BLUE JADE [17] to enact the service as an Agentcities node living on a J2EE server.
- All agents in the agency were built according to FIPA specifications.

Observe that Figure 1 shows the interplay of buying and providing agents with the Manager as the sole access point to the iBundler agency. In the following section, we make explicit the interplay of protocols involved in the whole interaction to compose the protocol of the service. Notice too that although so far we have referred to FIPA-compliant buyers and providers, in Section 3.5 we explain how buyers and providers can make available web façades for human agents to interact with them.

[1] Each number labelling an arc stands for the order of the message in the sequence of messages required by the negotiation protocol.

3.3. Interaction protocol

Figure 2 illustrates the interaction of a buying agent with iBundler when conducting a negotiation with a limited number of supplying agents. Notice that the interactions depicted among buying agents, iBundler, and supplying agents have been captured with the aid of the *Sniffer*[7] tool provided by *Jade*.

The visualisation depicted by *Jade Sniffer* is analogous to a UML sequence diagram, though the exchange of messages occurs between agents instead of objects. Furthermore, each interaction is assigned a step number corresponding to its position within the ordered sequence of messages it belongs to (as shown along the left hand side of the central picture).

The servicing of *iBundler* proceeds as follows. At step 1, the buying agent (labelled as *user* in Figure 2) sends his RFQ to the manager agent (*manager*) in the iBundler agency. At that point, the manager spawns an auxiliary agent, the so-called collector agent (*collector*), and delegates to him the collection of bids from supplying agents. Once created, the collector requests bids from supplying agents *provider1*, *provider2* and *provider3* by starting CFP (call for proposal) FIPA protocols (steps 2,3,4) that include the RFQ. Notice though that such CFP filters out the buying agent's constraints enclosed in the RFQ to hide them away from suppliers. These subsequently submit their bids to the collector via *propose* performatives (steps 5,6,7). Once bids are collected, the *manager* constructs the combinatorial problem to be solved from the RFQ submitted by the buying agent and all bids collected from suppliers. Furthermore, he agrees on providing an optimal solution to the buying agent (step 8). Next, the manager asks the translator agent (*translator*) for a solution to the combinatorial problem (step 9). Upon reception, the *translator* translates the combinatorial problem into an XML-based problem specification which is shipped to the *solver* component. The solution produced by the *solver* is returned to the *translator* as an XML-based document, so that it can be forwarded by the *translator* as a FIPA message to the *manager* (step 11). The optimal solution is finally sent over to the buying agent (step 12) so that he can employ it to decide which suppliers to award a contract, and for which items and units. The buyer's decision is made available to the *manager* (step 13), who requests the *collector* to award the contract to the selected providers, thus terminating the CFP protocol started out at step 2. Finally, the manager acknowledges the buyer that the contract has been indeed awarded to the providers, and on the terms he selected.

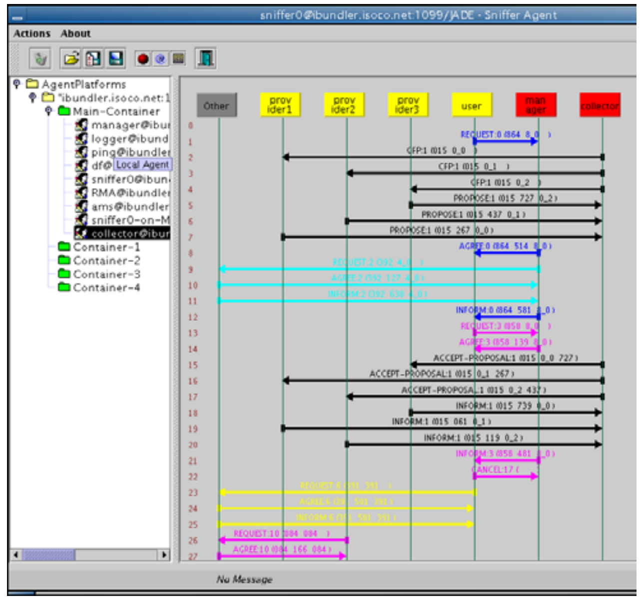

Figure 2. Interaction protocol between trading agents and the iBundler agency

Notice that the manager creates the collector to ease implementation ---since it is rather complex to extend the FIPA-compliant protocols offered by Jade---, to take advantage of FIPA-compliant protocols as offered by Jade, and to better control synchronisation over messages exchanged among all agents involved in the interaction with the iBundler service.

To summarise, the usage of the iBundler service requires the interleaving of several protocols, namely (agents involved in parentheses):

1. Request for optimal solution for an RFQ (buyer and manager – messages 1, 8, 12).
2. Collection of bids and awards (collector and suppliers – messages 2 to 7 and 15 to 20).
3. Request for optimal solution for RFQ and collected bids (manager and translator – messages 9, 10, 11).
4. Request for contract award (buyer and manager – messages 13, 14, 21, 22).

3.4. Ontology

Although research on automated negotiation in multiagent systems has concentrated on the design of negotiation protocols and their associated strategies, ontological aspects of negotiation protocols have recently started to attract researchers' attention [13]. In [13], we find an ontological approach to automated negotiation founded on the following concepts: *negotiation protocol* (rules followed by participants during a negotiation process), *party* (participants, be them either human agents, software agents or even organisations of agents), *process* (way to reach an agreement on some issue by modifying negotiation attributes), (negotiation) *object*, *offer* (possible combination of values associated to the negotiation attributes which represent an expression of will), *negotiation rule* (set of rules that govern a specific negotiation protocol). Although satisfactory enough for most concepts, particularly as to negotiation protocols and rules, in this work we had to enrich the concepts of *offer* and *object* in order to accommodate the expressiveness required by the actual-world constraints described in Section 2 for bids and RFQs respectively. To the best of our knowledge, no ontology defined in prior work allows us the expressiveness that buying and providing agents require. In other words, there is no adequate ontology for multi-item, multi-unit combinatorial reverse auctions with side constraints. Thus we had to define an *ad-hoc* ontology for the iBundler service.

The ontology has been defined with the aid of *Protege 2000* [16]. Furthermore, the conversion from ontological objects to Java classes is realised via the *beangenerator* Protege 2000 plug-in.

Figure 3 and Figure 4 provide graphical representations (as shown by the Ontoviz Protégé plug-in) of the core components of two central concepts in the iBundler ontology, namely and respectively the *RFQ* and *ProviderResponse* concepts.

The *RFQ* concept is employed by buying agents to express their requests for bids. Figure 3 shows that an RFQ is composed of a sequence of *Request* concepts, one per requested item. A sequence of global constraints (*GlobalConstraint* concept) relating separate, requested items may be part of an RFQ. There are two types of *GlobalConstraint* concepts: constraints that allow to express linear relationships between different attributes of the very same or separate item(s) (*AttributeRelation* concept) and constraints on the values of an item's attribute (*AttributeVariation* concept). A sequence of constraints on individual items (*RequestConstraint* concept) may be also part of an RFQ. Constraints on individual items can serve to limit the range of providers (*NumProviders* concept) to which the item can be awarded or the range of percentage of units to be awarded to the very same provider (*PerProvider* concept).

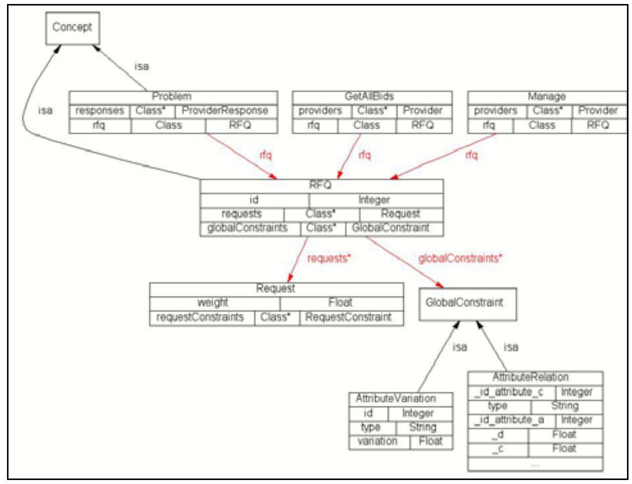

Figure 3. RFQ concept representation

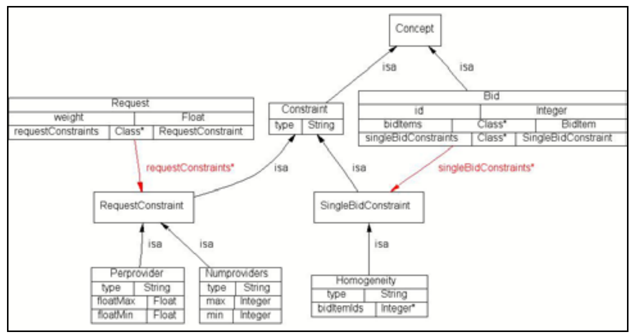

Figure 4. Bid representation as part of the ProviderResponse concept

On the provider side, providing agents express their offers in terms of the *ProviderResponse* concept, which in turn is composed of several elements: a list of *Bid* concepts (each *Bid* allows to express a bid per a single requested item or a bundle of items); constraints on the production/servicing capabilities of the bidding provider (*Capacity* concept); and constraints on bundles of bids formulated with the *BidConstraint* concept (each *BidConstraint* in turn can be of exclusive --xor-- or volume-based discount type --and--, corresponding respectively to the *XOR* and *AND* concepts). Whereas constraints on bundles of bids put into relation separate bids, constraints on individual bids (expressed as *SingleBidConstraint* concepts) allow to relate the values offered for separate items within the very same bid. As an example, homogeneity constraints can be declared by providers within some bid to make buyers aware that the quantity of items they can select per item must be the same, or else the provider will not concede his bid. Such constraint maps to the *Homogeneity* concept, a particular type of *SingleBidConstraint*.

3.5. Web façade

For illustratory purposes, we have also built web interfaces for people to interact with buying and providing agents. Figure 5 and Figure 6 show the web façades to a buying agent and to a limited set of providing agents. Notice that both façades offer a subset of the whole functionality available to buying and providing agents.

Figure 5 illustrates how to configure an RFQ to be delivered to a buying agent. The example shows the specification of a 5-item RFQ of some parts of the front suspension of a car. The buyer specifies the requested number of units, the percentage of contract to be allocated, the number of providers, and the degree of importance per item. The RFQ is shipped to a buying agent to initiate the negotiation protocol as shown in Figure 2.

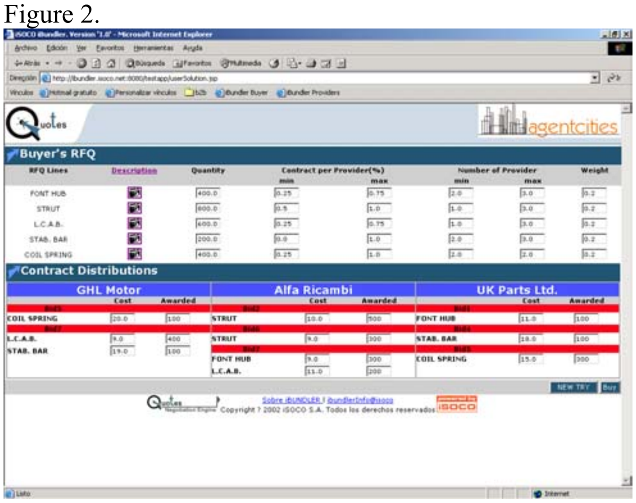

Figure 5. Web façade to a buying agent

Figure 6 depicts the interface for providers to compose both single and combinatorial bids. Bids express ranges of units in batches, and can be declared homogeneous or be related to be exclusive (xor bids) or inclusive (and bids). Once all bid values and their relationships are defined, they are handled to providing *agents. Figure 5* shows to the buyer (below *Contract Distributions)* the winning bids along with their costs as received by the buying agent from the manager agent.

4. Performance

Since combinatorial auction solvers are computationally intensive, a major issue is whether our service is to behave satisfactorily in highly-demanding trading scenarios. At this aim, we have conducted some empirical measures on the performance of iBundler Solver. Figure 7 shows how *Solver* behaves when solving negotiation problems as the number of bids, the number of items, and the items per bid increase. Notice that in order to run our tests, we devised a customisable generator of data sets which artificially created negotiation problems by wrapping the solution with noisy bids. In this way, not only were we able to measure the performance of Solver, but also to automatically verify the sound behaviour of the service in a large variety of negotiation scenarios demanding the many capabilities of the service.

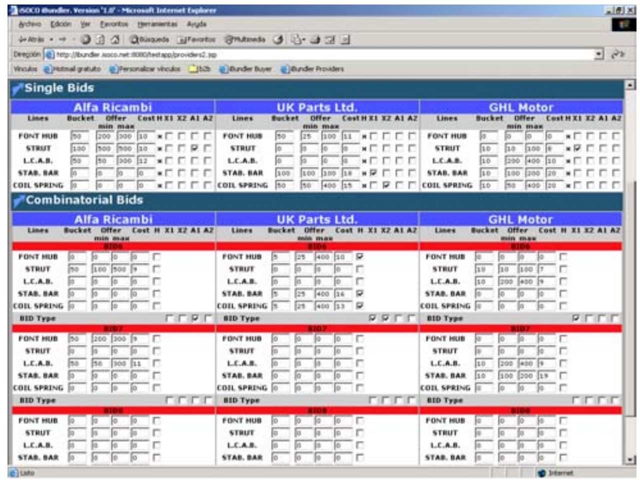

Figure 6. Web façade to providing agents

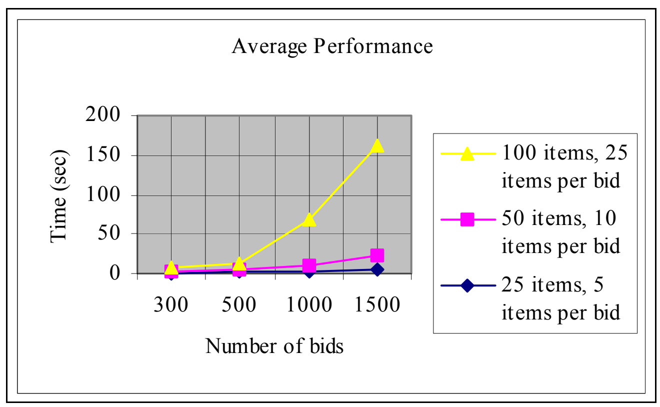

Figure 7. Average performance of iBundler

5. Conclusions and future work

This paper describes the iBundler service, a decision support agent for negotiation scenarios that operates as a combinatorial negotiation solver for both multi-item, multi-unit negotiations and auctions. iBundler was the winner of the Best Application of the 2003 Agentcities Technology Competition

Although negotiation is a key coordination mechanism, to the best of our knowledge only two Agentcities projects have concentrated on negotiation, namely ADMIT and Neg-Onto [18]. Both projects share the commonality of focusing on issues related to negotiation protocols and ontologies. Nonetheless, none of them provides decision support to buying and providing agents, and so both mainly focus on negotiation infrastructure issues. To the best of our knowledge the same observation applies to the agent services deployed over the DARPA Grid.

It is our view that the IBundler service attempts at making headway in the deployment of intelligent agent services that assist trading agents immersed in complex, actual-world negotiation scenarios. Therefore the IBundler service appears as an innovative decision support service for both negotiations and auctions that empowers agents to conduct from simple to largely sophisticated negotiations in open agent environments. We believe that it is time to deploy agent services aimed at complex problem-solving so that they can be subsequently employed by other agents either to help them team up and cooperatively solve complex problems or to behave more efficiently in competitive scenarios.

The iBundler service contributes along two main directions. On the one hand, we have incorporated new, actual side constraints to the winner determination problem for combinatorial auctions. On the other hand, we have defined a new ontology that accommodates both operational constraints and attribute-value constraints for buying and providing agents.

As to future work, we contemplate the development of a service that assists providing agents when bidding in complex, multi-item negotiation scenarios.

6. References

[1] Y. Fujishima, K. Leyton-Brown, Y. Shoham. "Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches". In Proceeding of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI'99, August, pp. 548–553.

[2] A. Reyes-Moro, J.A. Rodríguez-Aguilar, M. López-Sánchez, J. Cerquides, and D. Gutierrez-Magallanes. "Embedding decision support in e-sourcing tools: Quotes, a case study". *Group Decision and Negotiation Journal*, 2003 (to appear).

[3] M.H. Rothkopt, A. Pekec, and R.M. Harstad. "Computationally manageable combinatorial auctions". Management Science, 44(8):1131-1147, 1995.

[4] T. W. Sandholm. "Algorithm for optimal winner determination in combinatorial auctions". Artificial Intelligence Journal 135:1-54.

[5] T. W. Sandholm. "eMediator: A Next Generation Electronic Commerce Server". International Conference on Autonomous Agents (AGENTS), Barcelona, Spain, June 3-8 2000.

[6] S. de Vries and R. Vohra. "Combinatorial Auctions: A survey". January 12th, 2001.

[7] TILAB. "The Java Agent Development Framework". http://sharon.cselt.it/projects/jade/.

[8] E. Zurel, N. Nissam. "An Efficient Approximate Allocation Algorithm for Combinatorial Auctions". Proceedings of the ACM Conference on Electronic Commerce, Tampa, Florida, 2001.

[9] H.H. Hoos, C. Boutilier. "Solving Combinatorial Auctions using Stochastic Local Search". In Proceedings of AAAI-200, 2000.

[10] R.C. Holte. "Combinatorial Auctions, Knapsack Problems, and Hill-Climbing Search". Lecture Notes in Computer Science 2056.

[11] A. Andersson, M. Tenhunen, F. Ygge. "Integer programming for combinatorial auction winner determination". In Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS), Boston, MA, 2000, pp. 39-46.

[12] J. Collins, M. Gini. "An Integer Programming Formulation of the Bid Evaluation Problem for Coordinated Tasks". In B. Dietrich and R. V. Vohra, editors, Mathematics of the Internet: E-Auction and Markets, volume 127 of IMA Volumes in Mathematics and its Applications, pages 59-74. Springer-Verlag, New York, 2001.

[13] V. Tamma, M. Wooldridge, I. Dickinson. "An ontology for automated negotiation". In Proceedings of the First Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02), Bologna, Italy, 2002.

[14] T. Sandholm, S. Suri. "Side Constraints and Non-price Attributes in Markets". International Joint Conference on Artificial Intelligence (IJCAI), Workshop on Distributed Constraint Reasoning, Seattle, WA, August 4th, 2001.

[15] GNU. "GLPK – GNU Linear Programming Toolkit". http://www.gnu.org/directory/GNU/glpk.html.

[16] Protege 2000. http://protege.standford.edu.

[17] D. Cowan "The Blue Jade Project". http://sourceforge.net/projects/bluejade.

[18] Agentcities Project. http://www.agentcities.org.

[19] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. "Winner Determination in Combinatorial Auction Generalizations". First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02), Bologna, Italy, 2002, pp. 69-76.

[20] The Foundation for Intelligent Physical Agents. http://www.fipa.org.