Intelligent Agents for Automated One-to-Many e-Commerce Negotiation

Iyad Rahwan
Department of Information Systems
University of Melbourne
Parkville 3010, Australia
i.rahwan@pgrad.unimelb.edu.au

Ryszard Kowalczyk and Ha Hai Pham
CSIRO Mathematical and Information Sciences
713 Swanston Street, Carlton, Vic 3053, Australia
Ryszard.Kowalczyk@cmis.csiro.au, Ha.Pham@cmis.csiro.au

Abstract

Negotiation is a process in which two or more parties with different criteria, constraints, and preferences, jointly reach an agreement on the terms of a transaction. Many current automated negotiation systems support one-to-one negotiation. One-to-many negotiation has been mostly automated using various kinds of auction mechanisms, which have a number of limitations such as the lack of the ability to perform two-way communication of offers and counteroffers. Moreover, in auctions, there is no way of exercising different negotiation strategies with different opponents. Even though auction-based online trading is suitable for many applications, there are some in which there is a need for such greater flexibility. There has been a significant body of work towards sophisticated one-to-one automated negotiation. In this paper, we present a framework for one-to-many negotiation by means of conducting a number of concurrent coordinated one-to-one negotiations. In our framework, a number of agents, all working on behalf of one party, negotiate individually with other parties. After each negotiation cycle, these agents report back to a coordinating agent that evaluates how well each agent has done, and issues new instructions accordingly. Each individual agent conducts reasoning by using constraint-based techniques. We outline two levels of strategies that can be exercised on two levels, the individual negotiation level, and the coordination level. We also show that our one-to-many negotiation architecture can be directly used to support many-to-many negotiations. In our prototype Intelligent Trading Agency (ITA), agents autonomously negotiate multi- attribute terms of transactions in an e-commerce environment tested with a personal computer trading scenario.

Keywords: Automated Negotiation, Multi-Agent Systems, electronic commerce.

1 Introduction

E-commerce technologies have mainly been concerned with building an infrastructure for facilitating simple electronic document and fund transactions between enterprises. Apart from storing and exchanging information, and making bank transactions easier, faster and cheaper, e-commerce has done very little towards automating the way we do business. In particular, e-commerce has done little to automate decision-making processes humans typically get involved in as they conduct business transactions. Most existing e-commerce systems have been focused on the technical efficiency rather than the operational effectiveness in meeting high-level design objectives. In other words, these technologies are currently being used to reliably facilitate simple business transactions, falling short from supporting high-level automation of business decisions that underline better business efficiency and effectiveness. Simple matching, for example, takes place in most existing e-commerce environments due to its simplicity. However, these systems are far from meeting the needs of online traders. There is an obvious need for more advanced features that exploit the business potential available on the Internet, and its associated automation possibilities.

According to National Association of Purchasing Management estimates (NAPM), the average cycle time for setting up a master purchase agreement is 12 weeks. Current matching, cataloguing, and document and fund transfer technologies may reduce this cycle slightly, but the bulk of the time is wasted on the negotiation and conflict resolution processes. If we can automate the process of business negotiation in settings requiring personalised product delivery, the benefits will be tremendous. By cutting down the transaction cycle, and hence the transaction cost, we can achieve economies of scale.

As online trading becomes more common, a large number of electronic commerce services are being developed, which offer more sophisticated trading environments. Software agent technologies are promising great advantages to the way we do business (Jennings et al 2000), (Wurman 2001), (Rahwan, Kowalczyk and Yang 2001). In particular, systems that use software agent technologies are proving to be effective in helping users make better decisions when buying or selling over the Internet (Bailey and Bakos 1997). Software agents can also play an important role in providing automation and support for the negotiation stage of online trading (Guttman, Moukas and Maes 1997). Agent-based systems have been implemented, which support various stages of online trade; Namely: product brokering (determining what to buy), merchant brokering (determining who to buy from), and negotiation where all parties involved communicate in order to reach an agreement on the terms of transactions. An example of such systems is MIT’s Kasbah (Chavez and Maes 1996). More sophisticated automated trading systems have been proposed, which offer multi-attribute intelligent matching such as MIT's Tete-a-Tete (Guttman, Moukas and Maes 1997), and CSIRO’s ITA (Kowalczyk and Bui 2000). However, most of these systems support simple one-to-one automated negotiation between buyers and sellers. There are a number of technical and theoretical difficulties that need to be resolved before these systems realise their full potential. Moreover, the problem of supporting one-to-many negotiations is even harder.

Existing one-to-many negotiation systems rely mostly on rigid rules and are highly structured. They use economic and game theoretic techniques such as mechanism design (Kfir-Dahav 2000) in order to set up auctions that guarantee certain properties. Such settings have various advantages, but fail to support scenarios in which less structured, more flexible negotiation is needed. For example, agents lack the ability to perform two-way communication of offers and counteroffers. Moreover, in auctions, there is no way of exercising different negotiation strategies with different opponents. This can be beneficial when dealing with different markets, which exercise different negotiation rules, or even in settings where according to their performance history, different opponents may require different strategies.

In this paper, we present the latest version of our Intelligent Trading Agency (ITA), in which we attempt to achieve one-to-many negotiation by conducting a number of coordinated simultaneous one-to-one negotiations. The previous version (Kowalczyk and Bui 2000) was directed at facilitating one-to-one multi-attribute negotiation. In our current prototype, a number of agents, all working on behalf of one party, negotiate individually with other parties. Each agent conducts a direct negotiation with a prospective seller or buyer. After each negotiation cycle, these agents report back to a coordinating agent which evaluates how well each agent has done and issues new instructions accordingly. Each individual agent conducts its reasoning by using constraint-based techniques for evaluating and generating offers. The ITA agents autonomously negotiate multi-attribute terms of transactions in an e-commerce environment tested with the personal computer trading problem.

The paper is organised as follows. In the next section, we introduce our view of negotiation as a Distributed Constraint Satisfaction Problem (DCS). Then, in section 3, we introduce our mechanism for implementing one-to-many negotiations by instantiating and coordinating a number of parallel one-to-one negotiations. In section 4, we describe our simple prototype implementation of one-to-many negotiation for purchasing a personal computer. We state the major conclusions of our work and outline future research in section 5.

2 Negotiation as Distributed Constraint Satisfaction

Negotiation is a form of decision-making where two or more parties jointly explore possible solutions in order to reach a consensus (Rosenschein and Zlotkin 1994). In general, negotiation can be classified according to the number of parities involved and the number of attributes negotiated. In terms of the parties involved, negotiation scenarios can be one-to-one, one-to-many or many-to-many. In terms of negotiation attributes, a negotiation can involve single attribute (eg. price) or multiple attributes (eg. price, quality and delivery time). Our previous ITA system (Kowalczyk and Bui 2000) supported one-to-one multi-attribute negotiation. The current ITA supports multi-attribute, one-to-many negotiation.

We view negotiation as a process of cooperative and competitive decision making between self-interested agents in the presence of incomplete information. The agents have limited information about the preferences and constraints of each other. They make decisions according to available information about private preferences, constraints and individual negotiation strategies. The agents exchange information in the form of offers. An offer is a complete solution which is currently preferred by an agent given its preferences, constraints and the negotiation history of offers and counteroffers. An agreement takes place when a particular offer is accepted by all negotiation parties. During the negotiation process, the range of possible offers of each party changes according to the current information available. These ranges typically reduce to the final agreement, or if they become empty, a deal is not possible and the negotiation ends unsuccessfully. Therefore, negotiation is typically an iterative process of evaluating the offers, updating (eg. reducing) the available options, and making the counteroffers according to the individual negotiation strategies.

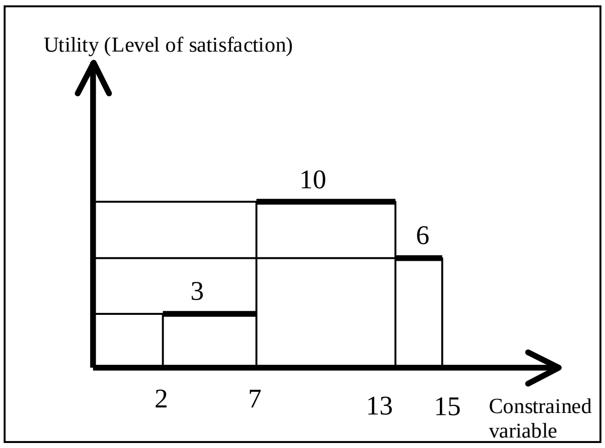


Figure 1: Constraint variable with different level of Satisfaction

The description mentioned above can be seen as a Constraint Satisfaction problem (CSP) (Kumar 1992) and in particular as a distributed CSP (DCSP) (Yokoo 2001), (Sycara et al 1991). CSPs are defined by a set of variables with associated domains, and a set of constraints on those variables. The objective is to find an instantiation of all variables while meeting all the constraints at the same time. A DCSP is a version of a CSP where variables are distributed among different agents. We should emphasise that, in the domain of competitive negotiation such as that in e-commerce, the exchange of information between agents is limited to the form of offers. All other information related to preferences, constraints, and offer evaluation and generation criteria of a particular agent are private and hidden from others.

Figure 1 shows how we use a combination of utility theory and constraints on domains of variables in order to specify more complex preferences. Using constraints with utility theory allows us to express constraints and preferences such as:

- The total price must be less than \$1000, but preferably less than \$900.
- The more warranty the better, but no matter how much more than 5 years, it doesn't make a difference.

Figure 1 may represent that the delivery date must be between 2nd and 15th of December. If the delivery date is between the 2nd and the 7th, our level of satisfaction is 3; we are most happy if it was between the 7th and the 13th; and a bit less happy if it was between the 13th and 15th.

2.1 Offer Evaluation

ITA agents use multi-attribute utility theory and constraint-based reasoning for the evaluation and generation of offers and counteroffers. Let us start with offer evaluation. For an offer received by agent A from agent B to be considered, it has to satisfy all constraints, that is, the proposed value of each variable must belong to its domain as specified by agent A (this domain is private information). Then, the value of an offer consisting of a number of attributes $x = \{x_1, \ldots, x_n\}$ is defined as a function

$$v(x_1,\ldots,x_n) = \sum_{i=1,\ldots,n} w_i v_i(x_i),\ \sum_{i=1,\ldots,n} w_i = 1$$

where x_i is the ith attribute of negotiation, v_i is a utility function of the ith attribute and w_i is the weight (priority) of the value of the ith attribute. The utility function is used for comparing and ordering alternative acceptable solutions. Figure 2 shows that the acceptable offer by both agents is one which satisfies at least the minimum utilities of both agents. If both areas of acceptability overlap, a solution can be found, otherwise, the negotiation fails. We can see the minimum boundary as a constraint on the domain of acceptable total utility values.

2.2 Offer Generation

Generating offers is the main decision making process that directs the progress of negotiation and its outcomes. It involves search for prospective solutions from the individual area of interest that move the parties towards an agreement from the common area of interest (Kowalczyk and Bui 2001) (see figure 2). ITA agents use constraint-based reasoning (constraint propagation) in order to search for a possible solution. This involves reducing the domain of variables (search space) to one that consists of feasible solutions satisfying the constraints of the party. Generating offers involves both constraint consistency maintenance and searching for the values of attributes to be offered. Constraint consistency maintenance (Yokoo 2001) involves posting new constraints as new information becomes available so they can be propagated during the search process in order to ensure the individual area of interest is consistent with the constraints of the party. This process takes place before and during the negotiation process. Searching for new offers involves selecting a particular value for the overall utility, and then performing constraint propagation in order to find the corresponding values of the attribute variables. Selecting new utility values is specified by the negotiation strategy. Since this paper is not in the scope of identifying different negotiation strategies, we have used a simple strategy in which the overall utility either stays the same (tradeoff) or reduces by a constant amount (concession). A tradeoff only takes place if there are different instantiations of attribute variables for the same total utility.

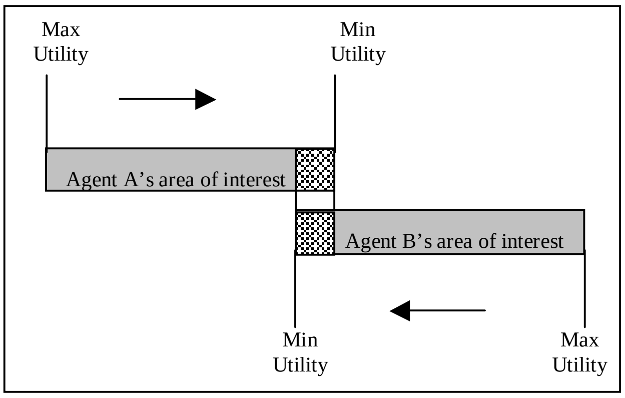


Figure 2. Constraint-based model of one-to-one negotiation

3 One-to-Many Negotiation Mechanism

The problem of automating one-to-many negotiation has proven to be hard (Parkes, Ungar and Foster 1999). This has lead to the wide use of highly structured one-to-many negotiation models based on auctions. Various types of traditional auctions, forward and reverse, are being used such as English, Dutch, and Vickery (Sandholm 1996). Although these negotiation models proved to be efficient and easily implementable in online applications, largely due to their simplicity, and although they fulfil the business needs in certain kinds of scenarios, they fail to support scenarios in which more complex, less structured negotiations occur. They follow a bidding style which considers competitive offers between participants flowing in one direction. A significant limitation of auction based negotiation systems is that they do not allow for interactive negotiation based on exchanging offers and counter offers, and thus exploiting the flow of information in both directions. With interactive negotiation, more information can be exchanged, and more flexible negotiation strategies become possible.

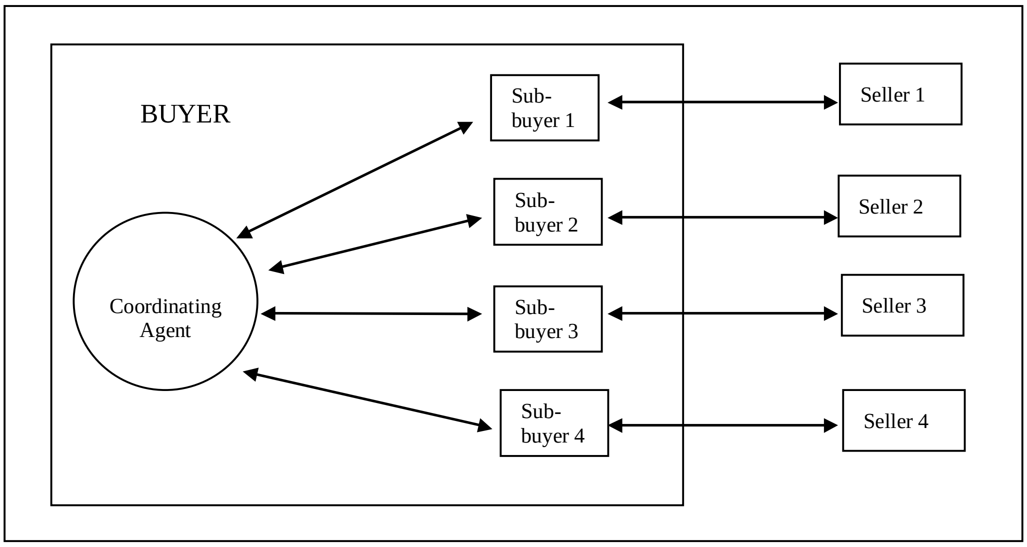


Figure 3. One-to-Many negotiation (One buyer & many sellers)

Moreover, having less structured negotiation rules means exercising different strategies with different opponents becomes possible, contrary to the case in auctions.

The first apparent technique towards automating flexible one-to-many negotiations is designing a complex, omniscient agent, which stores information about all current simultaneous negotiations at once in its state. However, this approach has a number of disadvantages. First, from a software engineering point of view, this approach poses a scalability problem, since our agent runs on one machine and may face problems trying to conduct an increasing number of concurrent negotiations. Secondly, adding or removing underlying one-to-one negotiation strategies requires rebuilding the agent. So this cannot be done at run time. There is an apparent need for more flexible, scalable, and reusable component based one-to-many negotiation. In this section, we describe our first step towards this goal.

We now present our approach to automating one-to-many negotiations. The scenario we are tackling is one in which one agent (buyer or seller) wants to negotiate a deal with a number of opponents, in order to find the best possible deal in the market. We propose reusing the techniques and components we used in one-to-one negotiations. This offers an advantage over approaches in which one single complex agent must conduct and directly maintain multiple threads of negotiation. In ITA, an agent can negotiate with many other agents by creating a number of one-to-one negotiating agents that negotiate on its behalf, and perform the task of coordinating them. We will call these agents *sub-negotiators*. Every sub-negotiator conducts a one-to-one negotiation with a different opponent. After each negotiation cycle (one offer and counteroffer), each sub-negotiator reports the results back to the coordinating agent. The coordinating agent then evaluates the situation, and issues instructions accordingly.

Figure 3 shows an instance of one-to-many negotiation scenarios. In this particular scenario, a buyer agent negotiates a deal with many prospective sellers. The buyer agent consists of a coordinating agent and a number of sub-negotiators (sub-buyers). All sub-buyers represent the preferences and constraints of the same buyer, but they may use different negotiation strategies. Similarly, a single selling agent can negotiate with a number of prospective buyers by instantiating a number of sub-sellers, and coordinating them.

From an architectural point of view, our approach has many advantages over existing systems:

– It offers simplicity and reusability by allowing us to reuse any existing one-to-one negotiating agent in a one-to-many setting, hence providing rapid development of negotiation applications.

– It allows for the system to be highly customisable since sub-negotiating agents can be modified, removed, or new agents with new strategies and capabilities can be added dynamically to the system at any point in time.

– This also allows for better scalability since not only can the different negotiation parties be on different machines over a network, but also can the different sub-negotiators of the same agent.

– The resulting system becomes more robust compared to a centralised complex agent. If one negotiation thread dies due to technical difficulties for example, the other threads can continue (as long as the coordinating agent is still alive).

– In principle, it would be also possible for each sub-negotiator to be a one-to-many negotiating agent consisting of a coordinator and several sub-negotiators, and so on.

In addition to the architectural advantages mentioned above, there are advantages relating to the method of representing preferences and constraints. We are using declarative knowledge representation in the form of constraints, which can be easily exchanged between, and understood by, different agents, makes adding and removing sub-negotiators and communication between them an easier task.

In ITA, there are two levels of negotiation strategies, namely strategies exercised by individual buyer or seller agents in their one-to-one encounters, and strategies exercised by the coordinating agents in organising and issuing commands to their sub-negotiators. Negotiation strategies of individual sub-negotiators include:

– Take it or leave it (fixed offer).

– No concession (same level of satisfaction, but possibly different offers).

– Fixed concession.

– Better deal strategies (resuming negotiation after deal is found hoping a better deal may come).

Details of these strategies and a few experiments on them can be found in (Kowalczyk and Bui 2001). We outline a few simple coordination strategies that can be exercised by the coordinating agent for controlling sub-negotiators:

a. **Desperate Strategy**: This is a very simple strategy in which the time constraints may be important and the agent wants to close a deal fast. In this strategy, as soon as a sub-negotiator finds an acceptable offer, the coordinating agent accepts it and sends messages to all other sub-negotiators to terminate their negotiation. If more than one sub-negotiator comes up with an acceptable offer, the one with the highest utility is chosen while the rest are also terminated.

b. **Patient Strategy**: In this strategy, even if an acceptable deal is found by one or more sub-negotiator(s), those agents are asked to wait while all other agents are asked to resume their negotiations. Once all sub-negotiators complete their negotiation process (whether with success or failure), the best offer is chosen. This strategy guarantees that the best possible deal can be reached, but does not give regard to time constraints. This might be a significant limitation in a marketplace with too many potential suppliers to negotiate with. One variation of the patient strategy is one in which a time limit is be set by the user, within which if no better deal was found, the negotiation terminates and the best deal so far wins.

c. **Optimised Patient Strategy**: In this strategy, the coordinating agent uses information about one negotiation outcome to influence the performance of other sub-negotiators. The constraints on the utility for the other sub-negotiators is updated in order to avoid unnecessary deals which are not as good as the one already found. For example, if the accepted minimum total utility is 5, and an one sub-negotiator has found a deal with utility 7, there is no point in other sub-negotiators reporting back a deal with utility 6 even though it is an acceptable deal (according to the initial constraints). In this case, the constraint on the utility for all remaining sub-negotiators is updated to be 7, causing any deal below that margin to be unacceptable. This also ensures that no sub-negotiator offers an offer that is worse than an offer received by a fellow sub-negotiator.

d. **Strategy Manipulation Strategies**: In this class of strategies, the coordinating agent may modify the negotiation strategies of different sub-negotiators at runtime. For example, after securing a deal, other sub-negotiators can exercise a take-it-or-leave-it strategy with their opponents. More sophisticated use of such strategies is left for future research.

One-to-Many Negotiation

One of the advantages of ITA is that they can automatically be extended to support many-to-many negotiations. This can be easily implemented by making each of the participants a one-to-many negotiating agent. Figure 4 shows an example of a many-to-many negotiations in which many buyers negotiate with many sellers. In this scenario, each self-interested buyer negotiates with many sellers in order to find the best deal, and each self-interested seller negotiates with many buyers in order to get the highest profit.

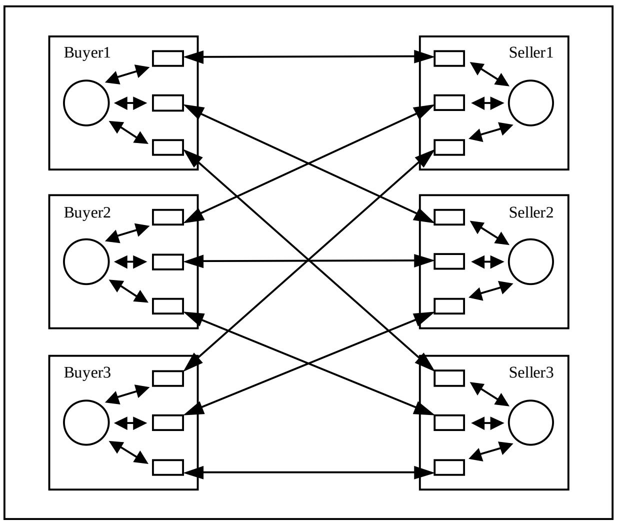
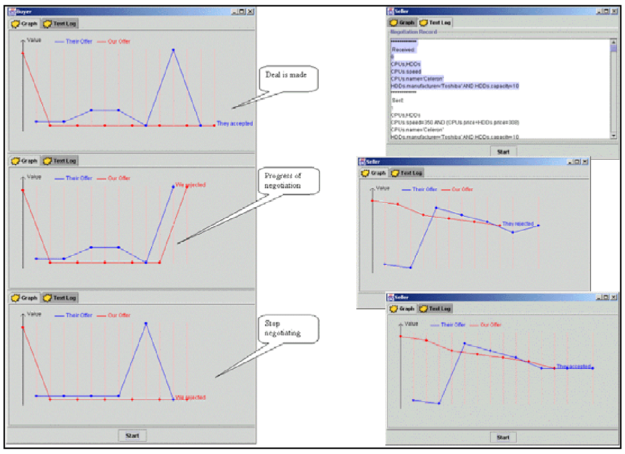


Figure 4. Many-to-many negotiation via multiple one-to-many negotiations

4 Prototype Implementation

In this section, we present a prototype implementation to demonstrate ITA's capabilities. We implemented our prototype using Java and allowing different agents to be located at different machines over the Internet. We used JSolver, a constraint programming library from Advanced Object Technology (Jsolver), to perform the constraint propagation and consistency maintenance. As a test bed, we used a PC trading scenario in which one buyer negotiates with three different sellers, each with different preferences and constraints. As opposed to the buyer which can generate any hypothetical offer that satisfies its constraints and preferences, the sellers are further constrained because the offers they generate are limited to those which they can actually provide. Each seller maintains a database of products available in the store.

In our system, the composite buyer agent consists of three instances of sub-buyers and a coordinating agent, implemented as a multi-threaded application. All sub-buyers are initialised with the same preferences and constraints, but may have different strategies. The user provides a configuration file, which represents the requirements (eg. monitor size, processor speed range, minimum memory, etc) and the shape of utility functions of different attributes (eg. different levels of satisfaction for different hard disk capacities). The user specifies two types of requirement specifications, those that are negotiable and those that are not. Non-negotiable attributes must be satisfied in any offer the seller generates. They are the only case in which information about constraints is publicly shared. Negotiable attributes, on the other hand, are those that the buyer has particular preferences about, which are not shared publicly. In the future, we intend to allow users to specify their preferences via a graphical interface as well as through configuration files.



```
************
Received:
0
CPUs,HDDs
CPUs.speed
CPUs.name='Celeron'
HDDs.manufacturer='Toshiba' AND HDDs.capacity=10
************
Sent:
1
CPUs,HDDs
CPUs.speed=350 AND (CPUs.price+HDDs.price=308)
CPUs.name='Celeron'
HDDs.manufacturer='Toshiba' AND HDDs.capacity=10
```

Figure 5. On the left, a buyer composed of three sub-buyers and the negotiation progression from their point of view, each with one independent seller.

A typical negotiation cycle is as follows. First, the coordinating agent creates and initialises a number of sub-buyers according to how many prospective sellers there are. Initially, all these buying agents are identical. Then, each pair of selling and buying agents can negotiate with each other simultaneously. A separate negotiation engine for each agent provides it with the main decision making functionality during a negotiation cycle, namely offer evaluation and generation. A negotiation cycle consists of one exchange of offers and counter offers by each pair. After each cycle, the sub-buyers report back their results to the coordinating agent. Since each sub-buyer negotiates with a different seller, after a few negotiation cycles, sub-buyers differ.

Figure 5 shows a screen shot of one buyer, composed of three sub-buyers, negotiating with three different sellers. The curves show the negotiation progression by each of the three different sub-buyers. The first sub-buyer managed to get an acceptable deal, while the others where asked by the coordinating agent to terminate the negotiation. Notice that the same negotiation may be viewed and recorded very differently by the negotiating buyer and seller. This is because they have different preferences and constraints and because they evaluate offers differently.

5 Conclusions and Further Research

This paper overviews an intelligent trading agency (ITA) to support fully autonomous multi-attribute one-to-many negotiations in the presence of limited common knowledge. We showed how a one-to-many negotiation could be implemented by coordinating a number of parallel one-to-one negotiations. Moreover, we showed how this could be easily extended to cater for many-to-many negotiation scenarios. We argued that our system offers advantages to the development of multi-agent negotiations, which can contribute towards simplicity, reusability, customisability, scalability and robustness. This is because sub-negotiating agents can be added or removed dynamically, for example when new agents with better strategies become available. This is facilitated by the fact that we are using declarative knowledge representation of preferences and constraints, which can be easily exchanged between, and understood by, different agents. Our individual agents negotiate multi-issue by exchanging offers and counter offers until they either reach a consensus that satisfies each party's private preferences and constraints, or they run out of offers and the negotiation fails. These agents use multi-attribute utility theory and constraint based reasoning for the evaluation and generation of offers. We implemented a pilot application that uses a PC trading scenario.

The coordinating agent is simple at this stage, providing a proof of concept. But the idea offers a very rich set of research issues. In the future, we would like to extend our work by implementing different coordination strategies, as well as study the dynamics of the system in more detail, by conducting a formal or empirical study. We believe there is a chance for exploring a large number of coordination strategies. For example, the coordinating agent may issue commands to change the sub-negotiators' individual strategies according to what is happening in the big picture. More sophisticated individual strategies and knowledge sharing between agents may open new possibilities. Another important area of research is the negotiation strategies of sub-negotiators. One direction could be investigating the use of learning techniques in order to allow agents to reuse their negotiation experience to improve the final outcomes.

6 References

Bailey, J. and Bakos, Y. (1997): An Exploratory Study of the Emerging Role of Electronic Intermediaries. *International Journal of Electronic Commerce*, 1(3), Spring 1997.

Chavez, A. and Maes, P. (1996): Kasbah: An Agent Marketplace for Buying and Selling Goods. *Proc. of the First International Conference on the Practical Appication of Intelligent Agents and Multi-Agent Technology*, London, UK, April 1996.

Guttman, R.H., Moukas, A.G. and Maes, P. (1998): Agent-mediated Electronic Commerce: A Survey. *Knowledge Engineering Review*, 13(3).

Jennings, N.R., Faratin, P., Norman, T.J., O'Brien, P., Odgers B. and Alty J.L. (2000): Implementing a

Business Process Management System using ADEPT: A Real-World Case Study. *Int. Journal of Applied Artificial Intelligence,* 14(5):421-465.

JSolver: http://www.aotl.com

Kfir-Dahav, N., Monderer, D. and Tennenholtz, M. (2000): Mechanism Design for Resource-Bounded Agents. *International Conference on Multi-Agent Systems (ICMAS),* Boston, MA, July 7-12, 2000.

Kowalczyk, R. and Bui, V. (2000): On Constraint-based Reasoning in e-Negotiation Agents. In *Agent Mediated Electronic Commerce III*. Dignum, F. and Cortés, U. (Eds). LectureNotes in Artificial Intelligence, Springer-Verlag, pp. 31 - 46.

Kumar, V. (1992): Algorithms for Constraint-Satisfaction Problems: A Survey. *AI Magazine,* Spring 1992, 32-44.

NAPM: www.napm.org

Parkes, D.C., Ungar, L.H. and Foster, D. (1999): Accounting for cognitive costs in on-line auction design. In *Agent Mediated Electronic Commerce*. Noriega, P. and Sierra, C. (eds). Lecture Notes in Artificial Intelligence 1571, 25-40, Springer-Verlag.

Rahwan I., Kowalczyk R., Yang Y. (2000): Virtual Enterprise Design – BDI Agents vs. Objects. In *Recent Advances in Artificial Intelligence in e-Commerce*. Kowalczyk, R. and Lee, M. (eds). Lecture Notes in Artificial Intelligence 2112, Springer-Verlag.

Rosenschein, J. and Zlotkin, G. (1994): *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. MIT Press.

Sandholm, T. (1996): Limitations of the Vickrey Auction in Computational Multiagent Systems. *Second International Conference on Multiagent Systems (ICMAS-96),* Keihanna Plaza, Kyoto, Japan, December, pp. 299-306.

Sycara, K., Roth, S., Sadeh, N. and Fox, M. (1991): Distributed constraint heuristic search. *IEEE Transaction on System, Man and Cybernetics* 21:1446-1461.

Wurman, P. (2001): Dynamic Pricing in the Virtual Marketplace (PR Wurman). *IEEE Internet Computing* 36-42, March/April 2001.

Yokoo, M. (2001): *Distributed Constraint Satisfaction*. Springer, 2001.