Modeling Complex Multi-Issue Negotiations Using Utility Graphs

Valentin Robu
robu@cwi.nl

D.J.A. Somefun
koye@cwi.nl

J.A. La Poutré
hlp@cwi.nl

CWI, Dutch National Research Center for Mathematics and Computer Science
Kruislaan 413, NL-1098 SJ Amsterdam, The Netherlands

ABSTRACT

This paper presents an agent strategy for complex bilateral negotiations over many issues with inter-dependent valuations. We use ideas inspired by graph theory and probabilistic influence networks to derive efficient heuristics for negotiations about multiple issues. Experimental results show — under relatively weak assumptions with respect to the structure of the utility functions – that the developed approach leads to Pareto-efficient outcomes. Moreover, Pareto-efficiency can be reached with few negotiation steps, because we explicitly model and utilize the underlying graphical structure of complex utility functions. Consequently, our approach is applicable to domains where reaching an efficient outcome in a limited amount of time is important. Furthermore, unlike other solutions for high-dimensional negotiations, the proposed approach does not require a mediator.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Intelligent Agents, Multiagent Systems; I.2.8 [**Problem Solving, Control Methods, and Search**]: Graph and tree search strategies, Heuristic methods

General Terms

Algorithms, Economics, Experimentation

Keywords

negotiation, market-based methods, utility graphs, influence diagrams, graphical models, game theory, decision theory

1. INTRODUCTION

Automated bilateral negotiation forms an important type of interaction in agent based systems for electronic commerce [9]. It allows seller and customer to determine the terms and content of the trade iteratively and bilaterally. Consequently, deals may be highly customized (especially for complex goods or services) and highly adaptable to changing circumstances. Moreover, by automating the negotiation, the potentially time-consuming process is delegated to autonomous software agents who conduct the actual negotiation on behalf of their owners.

In this paper, we consider the problem of a seller agent negotiating bilaterally with a customer about selecting a subset from a collection of goods or services, viz. the bundle, together with a price for that bundle. Thus, the bundle configuration — an array of bits, representing the presence or absence of each of the shop's goods and services in the bundle — together with a price for the bundle, form the negotiation issues. Like the work of [18, 6, 12, 16, 11], the techniques developed in this paper try to benefit from the so-called win-win opportunities, by finding mutually beneficial alternative bundles during negotiations.

The methods proposed in Faratin et al. [12], Coehoorn and Jennings [11] are geared towards finding win-win opportunities through modeling the preferences of the negotiation partner, for issues with independent valuations. This paper follows a similar approach; we consider *interdependencies* between issues, however, which make the problem considerably harder. In order to model such complex utility interdependencies between items, we introduce the novel concept of utility graphs.

Utility graphs build on the idea introduced in Chajewska and Koller [2] and Bacchus and Grove [1] that highly nonlinear utility functions, which are not decomposable in sub-utilities of individual items (such as in the seminal work of Raiffa [14]), may be decomposable in sub-utilities of clusters of inter-related items. They mirror the graphical models developed in (Bayesian) inference theory (cf. [7, 13]). Graphical models have been shown to be a powerful formalism for modeling decisions and preferences of other agents (see Lauritzen [7] for an overview). The idea behind using utility graphs in a multi-issue bargaining setting is to provide the seller with a formalism that can be used to explore the exponentially large bundle space, efficiently. In this paper, we show how utility graphs can be used to model an opponent's (i.e. customer's) preferences. Moreover, we also propose an updating procedure to obtain approximations of the customer's utility graph indirectly, by only observing his counter-offers during the negotiation.

An important benefit of using utility graphs in this setting is scalability: the problem becomes harder for larger number of items only if the underlying preference function becomes more complex as well. Graphical models (such as the one proposed in this paper), exploit the decomposable structure of utility functions, hence enabling more efficient search of the contract space.

At the start of a negotiation process, the seller's approximation of the customer's utility graph represents some prior information about the *maximal structure* of the utility space to be explored. This prior information could be obtained through a history of past negotiations or the input of domain experts. (An important advantage of a utility graphs is that they can handle both qualitative and quantita-

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

tive prior information.) After every (counter) offer of the customer, this approximation is refined. Conducted computer experiments show that — by using only a fairly weak assumption on the maximal structure of customers' utility functions — the updating procedure enables the seller to suggest offers that closely approximate Pareto efficiency [1]. Moreover, efficient outcomes are reached after relatively few negotiation rounds, which enables our approach to be used in applications where time constraints or the impatience of buyers are limiting factors.

The rest of this paper is organized as follows. Section 2 presents the negotiation setting: it defines the efficiency criteria, the negotiation protocol and the top-level outline of the negotiation algorithm. Section 3 defines the concept of utility graphs, it describes how such graphs can be used to model complex utility functions and to learn the preferences of the opponent in negotiation settings. Section 4 presents the set-up and results from the experiments performed to validate the model, while Section 5 concludes the paper with a discussion of related work and outlines the contributions of this paper.

2. THE NEGOTIATION SETTING

We consider a buyer and seller who negotiate bilaterally over a set of n binary-valued issues or items, and one continuous issue, the price. Henceforth, we will refer to the binary-valued issues as items and to subsets formed with these items as bundles. Negotiations are conducted in an alternating exchange of offers and counter offers. The offers and counter offers contain a n-dimensional vector of 0's and 1's representing an instantiation of the n items, plus a price offered/asked for this bundle.

The utility (measured in terms of monetary value) a buyer assigns to any bundle of items is given by a complex (nonlinear) function that takes into account interdependencies between various items. The seller's utility for a bundle (measured in net monetary value) is the difference between the price received for a bundle and the costs incurred for providing a bundle. These costs are additive: i.e., the bundle cost equals the cost of offering the items individually. Both the buyer's utility and the seller's cost represent private information, which remains undisclosed before or during the negotiation. Therefore, the negotiation setting can be describes as double-sided incomplete information.

2.1 Net Utility functions of Buyer and Seller

Let $B = \{I_1, \ldots, I_n\}$ denote the collection of n items a seller and buyer negotiate over. Each item I_i takes on either the value 0 or 1: 1 (0) means that the item is (not) purchased. Thus B has the domain $Dom(B) = \{0,1\}^n$ (so there are 2^n possible bundles). The n-dimensional vector $\vec{b} \in Dom(B)$ denotes an instantiation of these n items. In our approach, the utilities assigned to different outcomes (combinations) are represented by monetary units, rather than values between 0 and 1. We found this choice more natural for the e-commerce scenarios we consider. The utility function $u : Dom(B) \mapsto \mathbb{R}$ specifies the monetary value a buyer assigns to all (2^n) possible outcomes. Due to interdependencies between various items the function u is *highly* nonlinear. The buyer's *net utility* (i.e. net monetary value) for purchasing a bundle $\vec{b}$ for a price p, denoted by $nu_b(\vec{b}, p)$, is defined as follows:

$$nu_b(\vec{b}, p) = u(\vec{b}) - p. \quad (1)$$

That is, $nu_b(\vec{b}, p)$ is the difference between the monetary value for acquiring (consuming) bundle $\vec{b}$ minus the price p paid for purchasing bundle $\vec{b}$. The net monetary value of the seller is computed as:

$$nu_s(\vec{b}, p) = p - Costs(\vec{b}) \quad (2)$$

Thus, the seller's net monetary value for the sales of a bundle $\vec{b}$ for a price p is just the price minus the cost for selling the items. Currently, the seller has an *additive cost structure*: i.e., the bundle costs $Costs(\vec{b})$ equals the sum of the cost incurred when selling the items individually.

2.2 Using gains from trade as efficiency criteria

The most widely used performance criteria in multi-attribute negotiations is Pareto efficiency. Raiffa [14] provides a method to compute Pareto-efficient outcomes in the case utility functions of both parties are normalized between 0 and 1. In our model, utilities are represented in monetary units instead of mappings between 0 and 1. Consequently, it is more insightful to compute the gains from trade that can result from exchanging a certain bundle of items $\vec{b}$. The gains from trade are defined as:

$$GT(\vec{b}) = u(\vec{b}) - Cost(\vec{b}), \quad (3)$$

where $u(\vec{b})$ denotes the buyers monetary value for $\vec{b}$. The notion of gains from trade is well founded in the economic literature on trade (c.f. [10]). Moreover, for the above setting (where utility is expressed in monetary units) the set of bundles maximizing the gains from trade can be proven to be the same as the set of Pareto-efficient bundles (c.f. [16]). Intuitively, the gains from trade can be seen as the maximal size of joint gains, which can be achieved through negotiation, while the continuous attribute, the price represents different ways to divide these joint gains.

2.3 Top-level outline of the negotiation algorithm

The negotiation, in our model, follows an alternating offers protocol. At each negotiation step, each party (buyer/seller) makes an offer which contains an instantiation of 0/1 for all items in the negotiation set (denoting whether they are/are not included in the proposed bundle), as well as a price for this bundle. The decision process for each party, at each negotiation step, is composed of 3 inter-related parts: (1) take into account the previous offer made by the other party, (2) compute the contents (i.e. item configuration) of the next bundle to be proposed, and (3) compute the price to be proposed for this bundle.

Alg. 1 gives an outline of the algorithm the seller uses in each negotiation step. On the buyer's side, the followed steps mirror Alg.1 with the notably difference that the buyer does not try to estimate or reach the highest gains from trade, but selects the bundle that optimizes only his own expected net utility (i.e., he is entirely selfish). Based on the ongoing negotiation process he can update this choice, however. In this model, the burden of exploring the huge bundle space and recommending jointly profitable solutions is therefore passed to the seller, who must solve it by modeling the preferences of the buyer. This is a reasonable model in cases where an electronic merchant negotiates with different buyers in succession and tries to optimize both his own profits and buyer satisfaction (essential for building up a durable client relationship, which would generate repeated business). Furthermore, in e-commerce domains it is reasonable to assume that electronic merchants are more knowledgeable than individual buyers.

In our model the seller starts the negotiation by posting an ask price for each item. This price reflects the maximum profit the

[1] A deal is Pareto efficient whenever no party can be made better off without making the other one worse off.

Algorithm 1 Top level algorithm used by the seller

Denote by $(\vec{b}_b, p_b)$ the previous offer of the buyer and by $(\vec{b}_s, p_s)$ the previous offer of the seller.
1. If $\vec{b}_b = \vec{b}_s$ (configuration is agreed) and $p_s - p_b < \Delta p$ (difference in ask and offer prices is under some maximum acceptable threshold), then Success.
2. Otherwise:
3. Update the estimated utility graph of the buyer based on his past bid $\vec{b}_{buyer}$
4. Compute (one of) the bundles $\vec{b}^*$ with the highest gains from trade
5. Compute the price to be proposed for $\vec{b}^*$ such that it represents a linear time concession from my previous offer
6. Propose this bundle and price to the buyer

seller expects to make by selling that item (i.e. the ask price reflects his maximal aspiration level). The buyer also has a maximal aspiration level, which reflects the maximum discount he expects to get. The discount the buyer will actually get for a bundle depends on the content of that bundle and on the negotiation process itself.

The seller's approach is to search for a bundle that has the maximum gains from trade, since in this way he increases the joint gains: the only way he can continue to offer a better discount, but also increase his own profits, while the Buyer is strictly selfish and will always propose a bundle that increases his own (expected) utility. Regarding price concessions, both the buyer and the seller compute, for the current proposed bundle their current aspiration level and then make a time-dependent price concession with respect to that aspiration level.

For consistency, all experimental results reported in this paper refer to the *linear, time-dependent* concession case. However other time-dependent concession strategies usually employed in bilateral negotiations (such as hard-headed, boulware [5]) have been implemented and tested, with consistently good results. We note that the accuracy of the bundle predicted to have the highest gains from trade (step 3 and 4 in Alg. 1) does not significantly depend on the price concession strategy, because we made the explicit modeling choice that the concession strategy concerns only the pricing (step 5 of Alg. 1 above), not the exploration of the bundle content itself (steps 3 and 4 of Alg. 1). We acknowledge, however, that if the concession strategy were directly embedded in the opponent-learning mechanism (performed using the utility graph), the consistency of our experimental results may be effected. Finally, we model time pressure and/or buyer impatience through a break-off probability: at each step there is a small risk of breakdown (a value of 2% was used in the simulations).

3. USING GRAPHS TO MODEL UTILITY FUNCTIONS

3.1 Decomposable Utility Functions

Recall that we consider a buyer who negotiates with a seller over a bundle of n items, denoted by $B = \{I_1, \ldots, I_n\}$. Each item I_i takes on either the value 0 or 1: 1 (0) means that the item is (not) purchased. The utility function $u : Dom(B) \mapsto \mathbb{R}$ specifies the monetary value a buyer assigns to each of the 2^n possible bundles ($Dom(B) = \{0,1\}^n$).

In traditional multi-attribute utility theory, u would be decomposable as the sum of utilities over the individual issues (items) [14]. However, in this paper we follow the previous work of [2] by relaxing this assumption; they consider the case where u is decomposable in *sub-clusters* of individual items such that u is equal to the sum of the *sub-utilities* of different clusters.

DEFINITION 1. *Let C be a set of clusters of items $C_1, \ldots, C_r$ (with $C_i \subseteq B$). We say that a utility function is factored according to C if there exists functions $u_i : Dom(C_i) \mapsto \mathbb{R}$ ($i = 1, \ldots, r$ and $Dom(C_i) = \{0,1\}^{|C_i|}$) such that $u(\vec{b}) = \sum_i u_i(\vec{c}_i)$ where $\vec{b}$ is the assignment to the variables in B and $\vec{c}_i$ is the assignment to the variables in C_i, induced from the assignment $\vec{b}$. We call the functions u_i sub-utility functions.*

The factorization of a decomposable utility function is not unique. In this paper, we use the following factorization, which is a relatively natural choice within the context of negotiation. Single-item clusters ($|C_i| = 1$) represent the individual value of purchasing an item, regardless of whether other items are present in the same bundle. Clusters with more than one element ($|C_i| > 1$) represent the *synergy effect* of buying two or more items; these synergy effects are positive for complementary items and negative for substitutable ones.

The factorization defined above can be represented as an undirected graph $G = (V, E)$, where the vertices V represent the set of items I under negotiation. A vertex between two nodes (items) $i, j \in V$ is present in this graph if and only if there is some cluster C_k that contains both I_i and I_j. We will henceforth call such a graph G a *utility graph* [2].

EXAMPLE 1. *Let $B = \{I_1, I_2, I_3, I_4\}$ and $C = \{\{I_1\}, \{I_2\}, \{I_1, I_2\}, \{I_2, I_3\}, \{I_2, I_4\}\}$ such that u_i is the sub-utility function associated with cluster i ($i = 1, \ldots, 5$). Then the utility of purchasing, for instance, items I_1, I_2, and I_3 (i.e., $\vec{b} = (1,1,1,0)$) can be computed as follows: $u((1,1,1,0)) = u_1(1) + u_2(1) + u_3((1,1)) + u_4((1,1))$, where we use the fact that $u_5((1,0)) = 0$ (synergy effect only occur when two or more items are purchased). The utility graph of this factorization is depicted in Fig. 1.*

To give a numerical example, suppose: $u_1(1) = \$7$, $u_2(1) = \$5$, $u_3((1,1)) = -\$5$, $u_4((1,1) = \$4, u_5((1,1)) = \$4$. Moreover, all item costs are equal to \$2: i.e., $c(I_1) = c(I_2) = c(I_3) = c(I_4) = \$2$. In this case the bundle with the maximum gains from trade (i.e. the bundle denoted by $\vec{b}^*$ in Alg. 1) is: (0, 1, 1, 1), which has the net monetary value of \$5+\$4+\$4 - 3\$2 = \$7. From this simple example we can already highlight an important problem. The assignments for items I_1 and I_3, I_4 influence each other indirectly, through the assignment for I_2 (although there are no direct links from I_1 to I_3, I_4). This feature is exploited by our decomposition algorithm.*

At the computational level, each cluster is represented by a *joint utility table*, which assigns a utility value for all combinations of instantiations with 0/1 of items in that cluster (this is similar in concept to the joint probability tables, used to represent inter-dependent variables in probabilistic networks). Therefore, the cost of this utility representation is then exponential in the number of items in the cluster. For this reason, we limit the maximum size of any cluster to 4. However, this does not mean that the maximum size of the inter-dependencies is limited to 4 – it can be arbitrarily large. This is because one item can simultaneously belong to several clusters, so when deciding whether to include it or not in the optimal bundle (see section 3.3), the utility values in more than one table need to be

[2]The concept of cluster defined for utility graphs can be seen as corresponding to the concept of cliques in probabilistic networks. However, for consistency, we use only the term cluster throughout this paper.

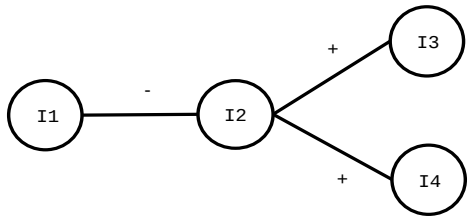

Figure 1: The utility graph that corresponds to the factorization according to C in Example 1. The $+$ and $-$ signs on the edges indicate whether the synergy effect are positive or negative.

taken into account. There is also a second, more fundamental reason to limit the maximum cluster size, i.e. with higher-order interdependencies, the decomposition algorithm on which our method is based is not guaranteed to produce a satisfactory partition (Section 3.3 gives a more precise description of this issue).

3.2 The maximal interdependency graph

As shown in the previous section, we assume that the buyer's utilities can be modeled as a graph. The structure of this graph, as well as the sub-utilities corresponding to different clusters constitute private information which is not revealed during the negotiation.

However, the seller does have some prior information to guide his opponent modeling. He starts the negotiation by having a *maximal item inter-dependence graph* of all possible inter-dependencies between the issues (items) which can be present in a given domain. The utility graphs of buyers form subgraphs of this graph. Note that this assumption says nothing about the weights or values of the sub-utility functions, so the negotiation is still with double-sided incomplete information. Furthermore, it does not mean that the seller has to know the exact structure of the utility graph of the buyer. For example, suppose two issues are assumed substitutable by the seller, so the utility of the combination containing both items is lower than the sums of utilities for individual items. If the buyer signals (through his bids) that he is willing to accept bundles containing both items, the seller will adjust the weight of this relation (i.e. adjust the values for the relevant cluster sub-utility) towards the sum of utilities for individual items. Conceptually, this is equivalent to removing the edge from the graph (which means they are no longer assumed substitutable).

The presence of this graph helps to greatly reduce the complexity of the search space on the side of the seller. The structural information contained in such a graph can be obtained either from a history of past negotiations or elicited from human experts. Note that in most domains it is reasonable to assume that the seller does know something about the goods he is selling. For example, if he is selling online pay-per-view journal articles, then articles within the same category (or with the same author) can be potentially related (though not guaranteed to be related for every buyer).

3.3 Selecting the best counter offer

A problem which the seller faces at each negotiation step (see Alg. 1, step 2) is to choose a bundle $\vec{b}^*$ which has the highest gains from trade of the 2^n bundles. More formally stated: $\vec{b}^* = argmax_{\vec{b}}(\hat{GT}(\vec{b})) = argmax_{\vec{b}}(\hat{u}(\vec{b}) - Cost(\vec{b}))$. Note that in the above definition, we use the notations $\hat{GT}$ and $\hat{u}$ instead of GT and u, since the seller does not know the *true* utility value of the buyer. He only has an estimation of it, which is updated after receiving a (counter) offer.

A straight-forward, brute-force solution to determine $\vec{b}^*$ is to generate all bundles $\vec{b}$ and select one which has the highest (estimated) gains from trade. Since this involves 2^n steps at *every* iteration, it is clearly not feasible for large n, so a heuristic is needed to reduce this search space.

Suppose the utility graph is decomposable into two or more completely disjoint parts (no overlapping vertices). We can then compute an optimal sub-bundle for each of the parts and merge them. However, for the more general case we still need a method for reducing the complexity of the search, when the original graph (or a large sub-component of it) is not decomposable in such a straight-forward way. We do this by applying ideas of the tree-decomposition theory to the utility graph $\hat{G} = (\hat{V}, \hat{E})$.

Informally, a tree decomposition of a graph is a family of small, not necessarily disjoint, subgraphs $\hat{G}_1, \ldots, \hat{G}_k$ (for some $k \in \mathbb{N}$), the union of which makes up the initial graph. Associated with a tree decomposition $\hat{G}_1, \ldots, \hat{G}_k$ is a collection of cutsets: a cutset is a subset of vertices that belong simultaneously to the same two subgraphs. (See [17] for a more formal discussion of tree decomposition algorithms).

EXAMPLE 2. *Figure 2 depicts such a decomposition. The vertices $\{Ia1, Ia2, Ia3, Ia4, Ic\}$ and $\{Ib1, Ib2, Ib3, Ic\}$ form the two cliques and corresponding subgraphs of $\hat{G}$. There are only two subgraphs therefore there is only 1 cutset; I_c forms this cutest because it is the only vertex that lies in both subgraphs.*

In the conducted experiments, cutsets contain at most 1 or 2 vertices. This is because, for any graph, polynomial-time algorithms exist for decomposing a graph according to cutsets of size 1 or 2, provided that the given graph can be decomposed in cutsets of size maximum 2 (cf. [17] for an overview). For the algorithm presented below, we do *not* need to decompose the whole graph, however. It suffices to decompose the graph at enough points to reduce its complexity. This means that the largest sub-component of vertices left (i.e. which is not or cannot be decomposed any further) has a manageable size to allow exhaustive search.

Algorithm 2 Algorithm returns $\vec{b}^*$, a bundle with the highest gains from trade (i.e., $\vec{b}^* \in \text{argmax}_{\vec{b} \in Dom(B)} \hat{GT}(\vec{b})$)

$\hat{G}_1 = (V_1, E_1), \ldots, \hat{G}_k = (V_k, E_k)$ and the union of all cutsets $S \subseteq V$ are given; $\hat{GT}_i : Dom(V_i) \mapsto \mathbb{R}$ denotes the predicted gains from trade resulting from the sales of a subset of the items in G_i; and $V_i[j], S[i] \in \{1, \ldots, n\}$ denote the reference to the item in B that corresponds to the j^{th} and i^{th}vertices in V_i and S, respectively.

1. $X := \emptyset$ //X will contain n-dimensional vectors
2. For all $\vec{s} \in Dom(S)$ {
3. Initialize $\vec{b}$ //$\vec{b}$ is a n-dimensional vector
4. For $(1 \leq i \leq k)$\{
5. //Get max gains from trade of G_i consistent with $\vec{s}$
6. $\vec{v}_i^* \in \text{argmax}_{\vec{x} \in Dom(V_i)} \hat{GT}_i(\vec{x})$
7. s.t. $\vec{x}(l) = \vec{s}(m)$ if $V_i[l] = S[m]$ for some $1 \leq l \leq |V_i|$ and $1 \leq m \leq |S|$
8. For $(1 \leq j \leq |V_i|)$ $\vec{b}(V_i[j]) := \vec{v}_i^*(j)$
9. \}$X := X \cup \vec{b}$
10.\} return $\vec{b}^* \in \text{argmax}_{\vec{x} \in X} \hat{GT}(\vec{x})$

Alg. 2 generates all possible combinations only for items that overlap between subgraphs (i.e. cutset items). Then, for all sub-

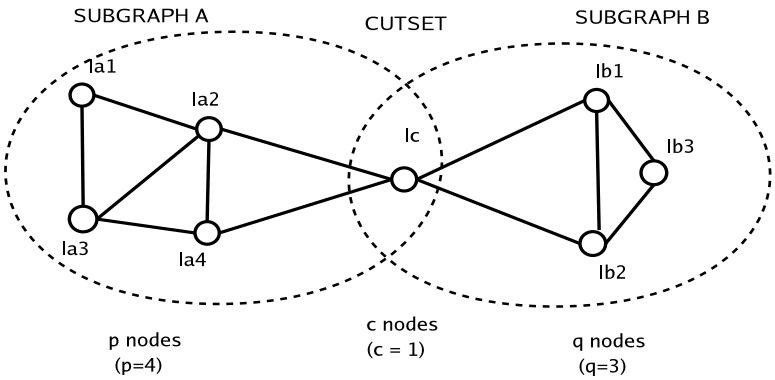

Figure 2: Utility graph where the vertices $\{Ia1, Ia2, Ia3, Ia4, Ic\}$ and $\{Ib1, Ib2, Ib3, Ic\}$ form the two cliques and corresponding subgraphs of $\hat{G}$. Moreover, $\{I_c\}$ forms the only cutest.

graphs it chooses the sub-combination that represents a *local* maximum for the gains from trade function in the considered subgraph, but subject to the constraint that the items that belong to more than one subgraph (i.e. cutset items) have the same values in all subgraphs. Finally, the best overall combination is chosen as a maximum of combinations of local maximums achieved for all possible instantiations for cutset vertices (items).

It can be shown that Algorithm 2 produces equivalent results to computing the bundle with the maximum gains from trade the straight-forward way, by generating all possible bundles. However, for large graph structures it is considerably faster (in fact the direct method is too slow to work in practice for large graph structures).

EXAMPLE 3. *Consider the graph in Figure 2. Generating all bundle combinations and testing them takes 2^{p+c+q} steps. Our algorithm generates all possible combinations only for cutset C, then computes optimal sub-bundles for subgraphs A and B for each combination of C and merges them. This requires only $2^c(2^p + 2^q)$ steps. Since c in our case is of size 1 or 2, while p and q can be arbitrarily large, the decrease in the number of steps is exponential.*

More generally, suppose the decomposition of a utility graph leads to k cutset nodes ($|S| = k$) and Max$V$ denotes the number of vertices of the subgraph with the largest number of vertices. The complexity of Algorithm 2 is then $O(2^{k+MaxV})$ (proofs are omitted due to lack of space).

3.4 Updating Sub-utility Functions

The search method described in Section 3.3 works on the utility graph $\hat{G}$, and corresponding sub-utility functions $\hat{u}_i$ ($i = 1, \ldots, |\hat{C}|$, where $\hat{C}$ denotes the set of clusters). They represent the best model of the opponent (i.e. buyer) that the seller has so far. After receiving a counter offer from the buyer, he will update this model. More precisely, he will update the sub-utility functions. In this Section, we will discuss the rule for updating these sub-utility functions.

The learning algorithm determines, for all $\hat{C}_i \in \hat{C}$, which combination in $\hat{C}_i$ was asked by the buyer at the last iteration (where there are $2^{|C_i|}$ possible combinations). Then it increases the expected monetary value of the buyer for that combination and degrades the other combinations in the cluster. Intuitively, the idea is to strengthen a link between vertices (represented by the corresponding sub-utility value) whenever a buyer indeed expresses an interest in purchasing the items corresponding to the vertices; otherwise the link is weakened. Algorithm 3 gives the actual updating rules.

Algorithm 3 Algorithm for updating sub-utility functions

The seller's and buyer's last offer contain the binary assignments $\vec{b}_s$ and $\vec{b}_b$ to the variables in B; moreover $\vec{c}_{i,s}$ and $\vec{c}_{i,b}$ denote the assignments to the items in $\hat{C}_i$, induced by $\vec{b}_s$ and $\vec{b}_b$ (i.e. $\vec{c}_{i,s}$ and $\vec{c}_{i,b}$ are sub-arrays of $\vec{b}_s$ and $\vec{b}_b$).

1. For $(1 \leq i \leq |\hat{C}|)$ {
2. if $\vec{c}_{i,s} \neq \vec{c}_{i,b}$ {
3. $u_i(\vec{c}_{i,b}) := u_i(\vec{c}_{i,b}) * (1 + \alpha_c)$
4. For all $\vec{c} \in \{0,1\}^{|C_i|} \setminus \{c_{i,b}\}$
5. $u_i(\vec{c}) := u_i(\vec{c}) * (1 - \alpha(i))\}\}$

EXAMPLE 4. *Suppose we have the cluster $C_i = \{I_3, I_5, I_6\}$ (for a $i \in \{1, \ldots, |C|\}$). The buyer's last offer contains the combination $I_3 = 0$, $I_5 = 1$, and $I_6 = 1$. Then the expected buyer utility for purchasing item 5 and 6 is increased: i.e., $u_i((0,1,1)) = u_i((0,1,1)) * (1 + \alpha(i))$. The expected utilities for all other combinations in $\{0,1\}^{|C_i|}$ (namely $u_i(1,1,0), u_i(1,0,1), u_i(1,0,0)$, etc.) are decreased.*

Parameter $\alpha(i)$ in Algorithm 3 defines how much weight should be given to the request from the buyer's last bid, in each cluster. A higher $\alpha(i)$ mean that the seller is more likely to give in to buyer's preferences for a cluster. A straightforward choice would be to assign the same α to all clusters. The seller would then only take into account the buyer's preferences. However, he should also take into account the expected gains from trade of a cluster, which also depend on his costs (unknown to the buyer). We therefore define a factor called the Gains from Trade Importance Ratio (GTR) as:

$$GTR(i) = \frac{GT_i(\vec{c}_{i,b}))}{GT(\vec{b}_b)} \quad (4)$$

for all $i = 1, \ldots, |C|$. In the above equation, $\vec{b}_b$ denotes the seller's last offer containing the assignment $\vec{b}_b$ to the variables in B and $c_{i,b}$ denotes the assignment to the variables in C_i, induced by $\vec{b}_b$.

Intuitively, this ratio provides a measure of the cluster's importance weight, by comparing the gains which can be obtained in this cluster to the gains for the whole bundle. The measure can be compared to the inverse of total number of clusters $1/|C|$, for the purpose of fine-tuning the alpha parameter for each cluster (although the clusters vary in size in the conducted experiments from 1-4 items, the above ratio still provides a useful heuristic). The

cluster-specific α is then computed as the sum between a fixed and variable component (computed by a sigmoid function): i.e.,

$$\alpha(i) = \alpha_{fixed} + \alpha_{var} * \frac{1}{1 + e^{\beta*(GTR(i)-1/|C|))}} \quad (5)$$

for all $i = 1, \ldots, |C|$. Here the parameter β is a positive value, which gives a measure of how steep the sigmoid function is. After conducting a number of experiments, we observed that transforming the function into a simpler step function (by assigning $\beta \to \infty$) produces reasonably good experimental results. The values assigned to the alphas in the conducted experiments are $\alpha_{fixed} = 0.1$ and $\alpha_{var} = 0.3$.

The rationale behind the above formula is the following: if a cluster has a high importance for the seller (i.e. if $GTR(i) > 1/|C|$), then the concession made for this particular cluster will be small (equal to α_{fixed}). Intuitively, this means that for this cluster, the costs of the seller are low, so the the seller should keep insisting more on offering his own values for the items in this cluster for longer, since he knows he can offer them cheaper (the buyer does not know this, because he does not know the cost structure of the seller). For clusters with relatively low gains from trade (i.e. if $GTR(i) < 1/|C|$), there is not much difference between the offer of the buyer and that of the seller - therefore the seller can concede towards the buyer's offer without much perceived utility loss.

We note that our updating rule implicitly entails that the seller uses a monotonic concession strategy, because in all clusters at least a small, non-negative concession α_{fixed} is made towards the buyer's offer. We made this choice since it assures that our algorithm guarantees convergence (i.e configuration agreement) within a limited number of negotiation steps (although how efficient this agreement is depends on the tuning of parameters).

We also note that our update model is geared to a particular class of buyer strategies, one in which the buyer will make an overall time-dependent concession at each step on either configuration or price. We can best exemplify this through the simplest strategy of this class: the case in which the buyer is "hard-headed" about the configuration, by insisting on her own values for all the items under negotiation, though he/she is willing to concede on the price. In this case, the seller will keep trying to offer the buyer other items, but will insist more on the items for which his costs are low. However, eventually she will concede by agreeing to the configuration asked (but she can still extract a considerable concession on the price).

4. EXPERIMENTAL ANALYSIS

4.1 Experimental set-up

There are several dimensions, which we want to test in our model:

- The distance to Pareto-efficiency of the outcome reached (measured, in our case, as the average percentage from the maximum possible gains from trade) as well as the robustness of this result to large variations in buyer and seller profiles.
- The time taken to reach a solution, measured as the number of negotiation steps.

We test our model in a negotiation over 50 binary-valued issues. The maximal preference graph, which the seller considers as possible, includes 28 clusters representing the direct synergy between items (either positive for complementarity or negative for substitutability effects): 4 of them contain 4 items each, 13 of them 3 items, 6 of them 2 items and 3 only 1 item. It is important to emphasize that in our representation, each cluster is represented through a utility table with one entry for all sub-combinations in the cluster (therefore of size $2^{|C|}$, where $|C|$ is the number of nodes per cluster). So, for example, for a cluster of size 3, we implicitly consider all its possible sub-clusters of size 2 and 1.

On the seller side, the cost of each of the n items is normally distributed according to $N(\mu_{cost}, \sigma_{cost})$. On the buyer side, the value of each of n individual items is normally distributed according to $N(\mu_{gains}, \sigma_{gains})$. Moreover, the synergy effects between *subsets* in each of the above 11 (non-singleton clusters) is normally distributed according to $N(0, k^2\sigma_{syn})$, where k denotes the number of items in the subset.

To somewhat limit the number of parameters, we set $\sigma = \sigma_{costs} = \sigma_{gains} = \sigma_{syn}$. The parameter σ captures the problem of finding Pareto-efficient solution very, nicely: the higher σ the higher the likelihood of complementarity and substitutability effects, hence the higher the likelihood of non-linearity, in the problem.

The mean of the cost distribution for each item μ_{costs} is always set to 1. For the mean of the distributions for the buyer μ_{gains}, we conduct experiment with 3 different values: 1.1, 1.25, 1.5. Otherwise stated, the valuations of the Buyer are, *on average* 10%, 25% to 50% greater than those of the Seller. In the reported tests, σ takes 8 values, ranging from 0 to 5. In other words, we consider to whole spectrum from no randomness, and consequently linear preferences, to a very high degree of randomness, and consequently (with probability) highly nonlinear preferences. Recall from Section 3.4 that the seller uses a subutility functions in conjunction with the utility graph to model a buyer. These functions are initialized such that the seller on average correctly predicts the opponent's utility. Already for relatively small values of σ the initially predicted utilities will (with all likelihood) be very different from the actual values, however.

For each combinations of values of μ_{costs}, μ_{gains}, and σ, 100 tests were performed. This means in total: $1 * 3 * 8 * 100 = 2400$ tests were performed (or 300 tests for each point σ reported in the figures below. For this number of tests per points the variance of the results does not decrease, we can therefore conclude that this number of tests provides statistically significant results.

4.2 Experimental results

The experimental results produced from the setting described above are given in Fig. 3 and 4. As mentioned, each point was produced from 300 negotiations and the error bars give the resulting variance. The two figures highlight the results with respect to reaching an agreement over the bundle configuration (i.e., the actual content of the bundle). By agreement, we mean that thereafter the bundle content no longer changes. After such an agreement is reached, it may take more negotiation rounds before bargainers agree upon the price (if an agreement is reached at all). The Pareto-efficiency of a deal — which is the focus of this paper — is then already determined, however.

There are several conclusions that can be drawn from the analysis of these results. Regarding the ability to find a bundle close to maximum efficiency (Fig. 3), we can conclude that our approach performs very well. (Please keep in mind that the Y axis in Fig. 3 is scaled for values between 80% and 100% of optimal). Even for very high values of σ, the algorithm is able to extract around 97% of the maximum gains from trade, with a variance of maximal 5%. This is particularly remarkable for very large values of σ (e.g., σ between 4-5). For these values, (with all likelihood) the problem becomes highly nonlinear and the buyer's utilities predicted initially by the seller will be very different from the actual values.

Regarding the number of steps needed to find the optimal configuration (Fig. 4), we do see a significant increase in this number for larger values of σ. Clearly, problems for larger σ are more difficult

to solve, so more steps are needed to correctly update the model of buyer's preferences and to find a good bundle. Nevertheless, a bundle very close to maximal efficiency is usually found, even in these more difficult cases.

Thus, the results strongly support the underlying idea put forward in this paper. That is, having a maximal utility graph of possible interdependencies can be used to successfully navigate the contract space and reach Pareto-efficiency with a limited number of steps, even for a relative large number of interdependent issues.

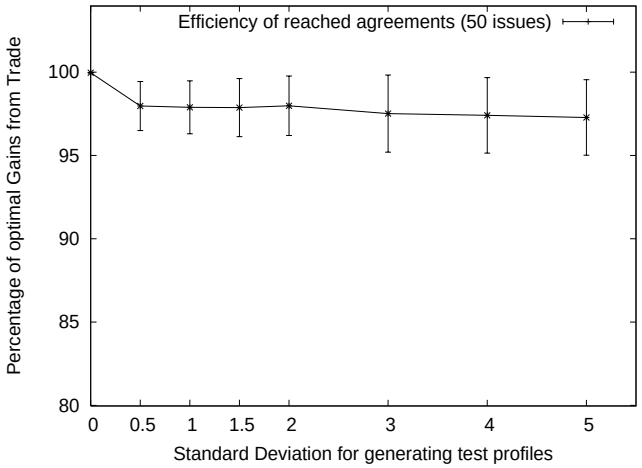

Figure 3: Percentage of the Gains from Trade (compared to the Pareto-optimal bundle) achieved over 100 tests

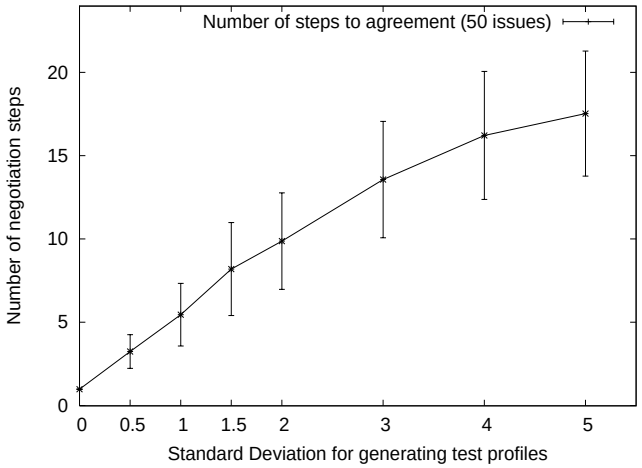

Figure 4: Number of steps needed to configuration agreement

5. DISCUSSION

In this section we provide a review of related work, with special attention to the features relevant for our approach. We conclude by summarizing the main contributions of our work and identifying directions for future research.

Most of the work in multi-issue negotiations has focused on the independent valuations case. Faratin, Sierra & Jennings [12] a method to search the utility space over multiple attributes is introduced, which uses fuzzy similarity criteria between attribute value labels as prior information. Coehoorn and Jennings [11] extends this model with a method to learn the preference weights that the opponent assigns to different issues in the negotiation set, by using kernel density estimation. Jonker and Robu [18] consider a similar model, the prior information are not fuzzy similarity criteria but a partial ordering of value labels. Both these papers have the advantage that they allow flexibility in modeling and deal with incomplete preference information supplied by the negotiation partner. They do not consider the question of functional interdependencies between issues, however.

Other approaches to multi-issue negotiation problem are the agenda based approach (Fatima et. al. [15]) and the constraint-based negotiation approach (Luo et. al. [19]). Both provide comprehensive formal analyses. They do not address the scalability and interdependence, however. Debenham [4] proposes a multi-issue bargaining strategy that models the iterative information gathering which takes place during the negotiation. Unlike our approach, the agents in [4] do not model the preferences of their opponent, but construct a probability distribution over all possible outcomes. However, this paper does not consider efficiency criteria for the reached outcomes (such as Pareto-efficiency), nor does it discuss scalability issues (it provides a convincing example, but restricted to only two issues).

Two negotiation approaches that specifically address the problem of complex inter-dependencies between multiple issues — and are therefore most related to our work — are [6, 8]. Klein et. al. [6] use a setting similar to the one considered in this paper, namely bilateral negotiations over a large number of boolean-valued issues with binary interdependencies (although we also allow higher-level interdependencies, up to 4 items). In this setting, they compare the performance of two search approaches: hill-climbing and simulated annealing and show that if both parties agree to use simulated annealing, then Pareto-efficient outcomes can be reached. In a similar line of work, Lin [8] uses evolutionary search techniques to reach optimal solutions. Both of these approaches have the advantage that they are scalable to large numbers of issues and Pareto-efficient outcomes can be reached without any prior information. However, a drawback of these learning techniques is the large number of negotiation steps needed to reach an agreement (around 2000 for 50 issues [6]).

By comparison with this work, our approach uses an explicit model of the buyer utility function - in the form of a utility graph. Although we assume no prior information about the preferences of any given opponent, we do assume that the maximal super-set of all possible interdependencies which are possible in the domain is known. This allows us to restrict the size of the search space and therefore reach Pareto-efficient agreements considerably faster. To some degree our approach could be also compared to solutions proposed to determine the winner in combinatorial auctions (Conitzer et. al. [3]). The problem is, in the general case, intractable, but by imposing some constraints on the type of bids which can be specified, efficient solutions can be found.

Our solution should be applicable in complex domains where fast agreements must be reached, by using some prior information about the structure of the utility space to be explored. This prior information could be a history of past negotiations or the input of domain experts. The relatively small number of steps needed to reach an agreement in our model allows it to be used in time-constrained negotiations or negotiation where impatience of one of the parties is a limiting factor. This is a significant result since, to the best of our knowledge, bargaining under time pressure in negotiations with many, inter-dependent issues was not considered in previous literature. Furthermore, in our setting, no trusted mediator is needed and the negotiating agents keep their preference information (i.e.

monetary utility, respectively costs) private.

Another direction of research relevant to our work is the one on graphical representations of utility functions. Bacchus and Grove [1] provide an early fundamental discussion of the graphical modeling of utility and preference functions and introduce the concept of conditionally additive independence. Their work is mostly concerned with formally analyzing the semantics and properties of graphical utility representations. Chajewska and Koller [2] build on this idea, by discussing the decomposability of non-linear utility functions into clusters and, in addition, they present an efficient procedure for eliciting such functions from human users, in a medical application domain.

In comparison with these works, our approach is geared towards finding an efficient algorithm which exploits the structure of utility graphs to model an online learning problem, such as negotiation, where agent preferences are revealed only indirectly, through repeated offers and counter-offers. Therefore we adapt the utility graph formalism for our setting, to assure a good trade-off between representation power of complex utility functions on one side, and computational efficiency in exploring the large bundle space on the other. To the best of our knowledge, very little work exists which applies this powerful formalism to settings which require online learning on the part of the agents, such as negotiation or virtual market settings.

Future work can follow two directions. A first direction for research should focus on using more advanced techniques inspired by learning in graphical models ([7, 13]), to construct the structure of utility graphs from scratch, without any prior information. For this purpose, aggregate customer information (for example information regarding all previous negotiations) could be used. A second direction is extending the current model to handle simultaneous one-one negotiation threads. For this, the problem of synchronizing the exchange of offers and especially the concessions made between parties remains an important open issue.

Another direction of potential further research is the applicability of the utility graph concept to model agent decision making in other settings. For example, in virtual market scenarios, utility graphs could be used to devise efficient bidding policies for simultaneous, sequential or repeated auctions.

6. REFERENCES

[1] F. Bacchus and A. Grove. Graphical models for preference and utility. In *Uncertainty in Artificial Intelligence UAI-95*, pages 3–10, 1995.

[2] U. Chajewska and D. Koller. Utilities as random variables: Density estimation and structure discovery. In *Proceedings of sixteenth Annual Conference on Uncertainty in Artificial Intelligence UAI-00*, pages 63–71, 2000.

[3] V. Conitzer, J. Derryberry, and T. Sandholm. Combinatorial auctions with structured item graphs. In *In Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2004.

[4] J. Debenham. Bargaining with information. In *3rd Int. Conf. on Autonomous Agents & Multi Agent Systems (AAMAS), New York, July 19-23, 2004*, pages 663–670, 2004.

[5] P Faratin, C Sierra, and N.R Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3–4):159–182, 1998.

[6] M. Klein, P. Faratin, H. Sayama, and Y. Bar-Yam. Negotiating complex contracts. *Group Decision and Negotiation*, 12:111–125, 2003.

[7] S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford UK, 1996.

[8] R. Lin. Bilateral multi-issue contract negotiation for task redistribution using a mediation service. In *Proc. Agent Mediated Electronic Commerce VI*, New York, USA, 2004.

[9] A. R. Lomuscio, M. Wooldridge, and N. R. Jennings. A classification scheme for negotiation in electronic commerce. *Group Decision and Negotiation*, 12:31–56, 2003.

[10] Andreu Mas-Collel, Michael D. Whinston, and Jerry R. Green. *Mircoeconomic Theory*. Oxford University Press, 1995.

[11] R. M. Coehoorn N. R. Jennings. Learning an opponent's preferences to make effective multi-issue negotiation tradeoffs. In *Proc. 6th Int Conf. on E-Commerce, Delft, The Netherlands*, pages 59–68, 2004.

[12] N. R. Jennings P. Faratin, C. Sierra. Using similarity criteria to make issue trade-offs in automated negotiations. *Journal of Artificial Intelligence*, 142(2):205–237, 2002.

[13] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Mateo, California, 1988.

[14] H. Raiffa. *The art and science of negotiation*. Harvard University Press, Cambridge, Massachussets USA, 1982.

[15] N. Jennings S. Fatima, M. Woolridge. Optimal negotiation of multiple issues in incomplete information settings. In *3rd Int. Conf. on Autonomous Agents & Multi Agent Systems (AAMAS), New York*, pages 1080–1087, 2004.

[16] D.J.A. Somefun, T.B. Klos, and J.A. La Poutré. Online learning of aggregate knowledge about nonlinear preferences applied to negotiating prices and bundles. In *Proc. 6th Int Conf. on E-Commerce, Delft*, pages 361–370, 2004.

[17] R. L. Rivest T. H. Cormen, C. E. Leiserson. *Introduction to algorithms*. The MIT Press, 1989.

[18] C. Jonker V. Robu. Automated multi-attribute negotiation with efficient use of incomplete preference information. In *3rd Int. Conf. on Autonomous Agents & Multi Agent Systems (AAMAS), New York*, pages 1056–1063, 2004.

[19] N. Shadbolt H. Leung J. H. Lee X. Luo, N. R. Jennings. A fuzzy constraint based model for bilateral multi-issue negotiations in semi-competitive environments. *Artificial Intelligence Journal*, 142 (1-2):53–102, 2003.