Genetic-Aided Multi-Issue Bilateral Bargaining for Complex Utility Functions

Tracking number: 167

ABSTRACT

In this paper, a multi-issue bilateral bargaining model is presented. It is based on a non-mediated protocol where agents are allowed to make up to k proposals each round. The strategy sends offers from the current iso-utility curve that are closer to the last offers sent by the opponent. The main aspect of this work is that it is designed to work with a large number of issues. Therefore, it is assumed that agents cannot explore their preferences completely and iso-utility curves cannot be calculated properly. A genetic algorithm (GA) is used to explore self-preferences. During the negotiation process, genetic operators are applied over the opponent's and one's own proposals in order to discover new proposals that are interesting for both parties. The results show that by using genetic operators better results are achieved. Moreover, the negative effect of introducing more issues is reduced by genetic operators, which makes the approach convenient for scenarios that have a large number of issues.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems, Intelligent agents*

General Terms

Negotiation algorithms

Keywords

Negotiation, Bilateral bargaining, Agreement technologies

1. INTRODUCTION

In the last few years, a new paradigm of "computing as interaction" has emerged due to the impact of new technologies (i.e., Internet, peer-to-peer, grid computing, etc). Under this paradigm, computing is something that is carried out through the communication between computational entities. In this sense, computing is an inherently social activity rather than a solitary one, leading to new forms of conceiving, designing, developing, and managing computational systems. The technology of agents/multiagent systems is particularly promising as a support for this new "computing as interaction" paradigm, and, most importantly, the development of techniques that enable software components to reach agreements (i.e., on the mutual performance of services). In fact, automated negotiation is highlighted as a core technology to reach agreements in this new computing paradigm [11]. Automated negotiation can be considered as an interaction and conflict resolution mechanism in situations where two or more parties have opposing preferences.

Most works in automated negotiation relate to multi-issue negotiation strategies where utility functions can be seen as a linear combination of the parameters involved in the negotiation process [1, 2, 3, 7]. Nevertheless, most real world domains present preferences that are much more complex than linear functions. In the last few years there has been an effort to research negotiation strategies that are capable of working with complex utility functions where issues may have relationships of interdependence. [5, 8, 16].

Works in these complex domains have focused on negotiation strategies that require a mediator [5, 8, 13]. Mediated strategies usually obtain very good results for all involved parties. However, a trusted mediator by all parties is not always possible in every real-world domain. Only a few works in the area have focused on providing non-mediated strategies in the complex preference case [10, 16]. Additionally, some strategies for complex utility functions often rely on the possibility of completely exploring agent preferences [10]. Unfortunately this may not be feasible, especially in cases where there are a large number of negotiation issues and agent preferences may frequently change.

In this work, a non-mediated bilateral multi-issue negotiation model where agent preferences are private is presented. The developed strategy is independent of the underlying complex utility function due to a heuristic that is similar to the one proposed in Lai et al. [10]. Both protocols are based on the Rubinstein alternating protocol [15] where each agent can propose up to k different proposals in each round. The main difference between the two approaches resides in the fact that in our work it is assumed that agents are not capable of exploring completely their preferences due to a large number of issues. Consequently, iso-utility curves cannot be calculated properly. A genetic algorithm (GA) is employed by each agent before the negotiation process in order to determine its own good proposals. During the negotiation process, each agent applies genetic operators over received proposals and their own proposals. The goal is to perfom an additional exploration that takes into account both agent preferences in order to obtain proposals that are interesting to both parties. The proposals

Cite as: Genetic-Aided Multi-Issue Bilateral Bargaining for Complex Utility Functions, Author(s), *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

are sent from the current iso-utility curve. These proposals are the k most similar proposals to the last received offers. They come from proposals that have been explored before the negotiation process as well as new proposals that have been explored using genetic operators. The results show that the use of genetic operators during the negotiation process leads to better results in joint utility, distance to Nash equilibrium, distance to closest Pareto optimal point, and number of negotiation rounds. More importantly, it is also shown that the use of genetic operators greatly reduces the impact of working with a large number of issues, which is considerably important in real-world domains.

This paper is organized as follows. Section 2 describes the negotiation model, explaining the chosen protocol and the new negotiation strategy in detail. In Section 3, the experimental setting and the results obtained are discussed. In Section 4 related work is be discussed. Finally, the conclusions and future lines of work are explained in Section 5.

2. NEGOTIATION MODEL

A negotiation model is formed by a negotiation protocol and a negotiation strategy. The negotiation protocol defines the rules to be followed by participant agents in the negotiation process. The protocol specifies which actions are allowed, their order, which agent performs each action, and so forth. The negotiation strategy defines the decision-making mechanism for a participant agent in a negotiation that is governed by that particular protocol. Both the negotiation protocol and the new negotiation strategy, are be described below.

2.1 Negotiation Protocol

The employed protocol is the one proposed by Lai et al. [10]. It is a bilateral bargaining protocol that is based on the Rubinstein alternating protocol [15]. Two different agents negotiate about a set of issues in an ordered way. First, an agent can make up to k different offers. The other agent may accept one of the offers received or reject all of them. If the agent accepts, the negotiation process ends with an agreement by both parties. However, if the offers are rejected, the other agent can propose up to k different offers that will be sent back to the first agent. The first agent will decide wheter or not to accept one of the offers, either reaching an agreement or rejecting all of them. This process is repeated until an agreement is reached by both parties or one of the agent decides to abandon the process.

2.2 Negotiation Strategy

The new proposed strategy can be described according to three different aspects. The first aspect relates to self-preference exploration and is carried out before the negotiation process itself. In this part agent preferences are explored by means of a GA since a complete exploration would result in a process that is too costly (i.e., negotiations about many issues). The second aspect is the evaluation of proposals, acceptance criteria, concession strategy, and so forth. Finally, the third aspect relates to how to select offers that are to be sent to the opponent. Genetic operators are carried out over received offers and the agent's own proposals in order to explore new proposals that may be of interest to both parties. A brief outline of the proposed strategy can be observed in Algorithm 1. A more detailed outline of the strategy used during the negotiation process can be observed in Algorithm 3.

Algorithm 1 A brief outline of the negotiation strategy

Negotiation Strategy
1.Explore self-preferences
2.Start negotiation process
3.Receive opponent offer(s)/ counteroffer(s)
4.Evaluate offers: Accept and go to step 9, or reject and continue the negotiation process
5.Explore new proposals using genetic operators
6.Select offer(s)
7.Send offer(s)
8.Go to step 3
9.End

2.2.1 Pre-negotiation: Explore self-preferences

When an agent uses complex utility functions to represent its preferences it may find complex distributions for good proposals. Nevertheless, when the number of issues is large, the complete exploration of agent preferences can be a computationally expensive process. For instance, exploring a negotiation domain formed by 10 integer issues from 0 to 9 requires exploring 10^{10} proposals. The cost associated to this exploration can be exhorbitant, especially if agent preferences change with a frequency that is greater than the time invested in preference exploration. The exploration process can be reduced by skipping proposals that are of very low quality for the agent (i.e., proposals with utility equal to zero).

A possible solution to this problem is to use mechanisms that allow us to find good proposals for the negotiation process and skip low quality ones. In this work, a GA was used to solve this problem. GA's are general search and optimization mechanisms based on the darwinian selection process for species [4, 6]. Genetic operators such as crossover, mutation, and selection are employed in order to find near-optimal solutions for the required problem. Nevertheless, classic GA's pose the problem that the entire population converges to one optimal solution. In this work, different interesting proposals for the negotiation process need to be explored. Niching methods are introduced to confront problems of this kind [12, 14]. These methods look to converge to multiple, highly fit, and significantly different solutions.

A possible family of niching methods for GA's is the crowding approach [14]. Crowding methods achieve the desired result by introducing local competition among similar individuals. One advantage of crowding methods is that they do not require parameters beyond the classic GA's. Euclidean distance is usually used to assess the similarity among individuals. Probabilistic Crowding (P_C) and Deterministic Crowding (D_C) [14] are two of the most popular crowding methods. They only require a special selection rule with respect to classic GA's. Both rules are employed to select a winner given n different individuals. On one hand, D_C selects the individual that has the highest fitness value, resulting in an elitist selection strategy. On the other hand, in P_C, there is certain probability for lower fitness value individuals to be selected as winners. This probability is usually proportional to the fitness of each individual. P_C behaviour is more exploratory than D_C. In both cases, the niching effect is achieved by applying either of the two rules to those individuals that are similar. Each parent is usually paired with one of its children in such a way that the sum of the distances between pair elements is minimum. For each pair, one of the two crowding rules is employed to determine which individuals will form the next generation. D_C and P_C can be observed in more detail in Equations 1 and 2, respectively.

$$D_c(s_1, s_2) = \begin{cases} s_1 & f(s_1) > f(s_2) \\ s_2 & f(s_1) < f(s_2) \\ s_1 \vee s_2 & \text{other} \end{cases} \tag{1}$$

$$P_c(s_1, s_2) = \begin{cases} s_1 & f(s_1) > f(s_2) \ \wedge \ rand \leq p_1 \\ s_2 & f(s_1) > f(s_2) \ \wedge \ rand > p_2 \\ s_2 & f(s_1) < f(s_2) \ \wedge \ rand \leq p_2 \\ s_1 & f(s_2) < f(s_1) \ \wedge \ rand > p_1 \\ s_1 \vee s_2 & other \end{cases} \tag{2}$$

$$\text{with } p_i = \frac{f(s_i)}{f(s_i) + f(s_{i'})}$$

where $rand \in [0, 1]$, $f(.)$ is the fitness function, s_1 and s_2 are two solutions, and p_1 and p_2 are the probability of acceptance of both solutions given the pair (s_1, s_2).

The designed solution uses crowding methods in GA's to find different good proposals. This GA is individually executed by each agent before the negotiation process begins. In this work, negotiation issues are in the integer domain. Nevertheless, the devised strategy is easily adaptable to new domains by making small changes in the genetic operators. The chromosome representation selected is an integer array where each position represent a negotiation issue. A portfolio with D_C and P_C is used. The population has a fixed number of individuals. The whole population is selected to form part of the genetic operator pool. Pairs of parents are selected randomly and multi-point crossover or mutation operators are applied over them. In both cases, the result is two children. Each parent is paired with the child that is more similar to it according to euclidean distance. P_C or D_C is applied to each of the pairs. Selected individuals replace the current generation. The stop criterion was set to a specific number of generations. At the end of the process, the whole population should have converged to different good proposals that are to be used by the negotiation process as an approximation to the real preferences of the agent (P). A more detailed outline of the proposed GA can be observed in Algorithm 2.

2.2.2 *Evaluate offers*

The negotiation strategy is time-dependent. In each negotiation round, the agents concede until a private deadline is reached. The utility of an agent a for an instant t of the negotiation process can be formalized as follows:

$$U_a(t) = 1 - (1 - RU_a)(\frac{t}{T_a}) \pm \delta \tag{3}$$

where $U_a(t)$ is the current utility level for agent a at instant t, RU_a is the reservation utility, T_a is the private deadline, and δ is a small value that makes the mechanisms less strict with respect to the current utility level. For instance, if an agent's current utility is set to 0.7, and each step the agent concedes 0.1, then a small value of $\delta = 0.01$ allows proposals that are between 0.69 and 0.71 to be accepted or sent.

Algorithm 2 Pre-negotiation: Genetic algorithm with niching mechanism. Its goal is to approximately explore agent preferences and obtain good quality proposals

```
P : Explored preferences, good quality proposals
D_c : Deterministic crowding rule
P_c : Probabilistic crowding rule
p_cr : Probability of crossover operator
p_dc : Probability of DC
n : Current number of generations
n_max : Maximum number of generations
pair_i : Pair of solutions

Initialize P
n = 0

Do
    n = n + 1
    shuffle P
    P_aux = ∅
    i = 1
    While i ≤ |P| − 1
        p_1 = P_i
        p_2 = P_{i+1}
        If Random() ≤ p_cr
            (c_1, c_2) = crossover(p_1, p_2)
        Else
            c_1 = mutate(p_1)
            c_2 = mutate(p_2)
        EndIf
        (pair_1, pair_2) = argmin_{p_i ≠ p_j, c_k ≠ c_l} ||p_i − c_k|| + ||p_j − c_l||
        If Random() ≤ p_dc
            Add(P_aux, D_c(pair_1))
            Add(P_aux, D_c(pair_2))
        Else
            Add(P_aux, P_c(pair_1))
            Add(P_aux, P_c(pair_2))
        EndIf
        i = i + 2
    EndWhile
    P = P_aux
While n ≤ n_max
Return P
```

Given the set of offers $X^t_{b\to a}$ received by agent a from agent b at instant t, the mechanism for accepting proposals of agent a can be formalized as depicted in the following expression:

$$Accept^t_a(X^t_{b\to a}) = \begin{cases} \text{accept} & V_a(x^{t,best}_{b\to a}) \geq \underline{U_a(t+1)} \\ \text{reject} & otherwise \end{cases} \quad (4)$$

where $Accept^t_a(X^t_{b\to a})$ is the offer acceptance function, $V_a(x)$ valuates the utility of a proposal, $x^{t,best}_{b\to a}$ is the best offer received from the opponent at instant t, and $\underline{U_a(t+1)}$ returns the lower bound for the interval defined by $\overline{U_a(t+1)}$.

2.2.3 Propose new offers

The next step in specifying the negotiation strategy consists of defining the mechanism to propose new offers. It is neccessary to devise a mechanism that is capable of proposing up to k different offers to the opponent. The chosen heuristic is similar to the inspiring work of Lai et al. [10]. Nevertheless, in this present work there are two main differences. First, the applied heuristic takes into account the k proposals received from the opponent. It does not limit itself to work only with the highest utility offer received from the opponent. Second, it is only possible to access the approximation an agent's preferences (P) explored by the first GA. Even though, it is not possible to access the full iso-utility curve, an approximation of it can be accessed. Genetic operators are used over received proposals and proposals that are the most similar from current iso-utility curve in order to explore new proposals that might be of interest to both parties.

Explore new regions by genetic operators.

One of the features of this work is the exploration of new proposals during the negotiation process by means of genetic operators. When performing genetic operators over proposals that come from the opponent and one's own proposals, the effect is that the resulting children have genetic material from both proposals. The hypothesis is that new offers generated by this mechanism have good utility for both parties. The heuristic looks for new *win-win* situations by means of genetic operators.

Let us consider $X^t_{b\to a} = [x^{t,1}_{b\to a}, x^{t,2}_{b\to a}, ..., x^{t,k}_{b\to a}]$, which is the set of proposals sent by agent b to agent a at instant t. For each offer $x^{t,i}_{b\to a}$, a total of M offers are selected from the current iso-utility curve defined by the first exploration process P. These M offers minimize the expression:

$$\underset{|C|=M}{\text{argmin}} \sum_{j=1}^{M} ||x^{t,i}_{b\to a} - c_j|| \quad (5)$$

where C is the set of M different offers, and $||x^{t,i}_{b\to a} - c_j||$ is the euclidean distance between one of the offers in C and the offer received from the opponent.

Once the M closest offers have been selected, a total of n_{cross} crossover operations are performed for each pair $(x^{t,i}_{b\to a}, c_j)$. The crossover operator takes two parents and generates one child. More specifically, the number of issues that come from $x^{t,i}_{b\to a}$ is chosen randomly from 1 and $N-1$, with N being the number of issues. The rest of the issues come from c_j. Which issues come from each parent is also decided randonmly. Each child is added to a special pool

---

Algorithm 3 Negotiation strategy during the negotiation process

P: Proposals from first GA
P_{new}: New proposals discovered
k: Number of proposals of the protocol
M: Number of selected offers
n_{cross}: Number of times to crossover
n_{mut}: Number of times to mutate
p_{pnew}: Proportion of proposals from P_{new}

/*Receive offers and evaluate offers*/

Receive $X^t_{b\to a}$

$x^{t,best}_{b\to a} = \text{argmax}\, V_a(X^t_{b\to a})$

If $V_a(x^{t,best}_{b\to a}) \geq \underline{U_a(t+1)}$ then Accept

Update current utility

/*Propose new offers*/

/*Explore new proposals by genetic operators*/

For each $x^{t,i}_{b\to a}$ in $X^t_{b\to a}$
    $C = \underset{\substack{C\subset P \\ |C|=M}}{\text{argmin}} \sum_{j=1}^{M} ||x^{t,i}_{b\to a} - c_j||$
    For each c_j in C
        Repeat n_{cross} times
            s1=Crossover($x^{t,i}_{b\to a}, c_j$)
            If s1 $\not\subseteq P_{new}$ then Add(P_{new},s1)
            Repeat n_{mut} times
                s2=Mutate(s1)
                If s2 $\not\subseteq P_{new}$ then Add(P_{new},s2)
            EndRepeat
        EndRepeat
    EndFor
    Repeat n_{mut} times
        s1=Mutate($x^{t,i}_{b\to a}$)
        If s1 $\not\subseteq P_{new}$ then Add(P_{new},s1)
    EndRepeat
EndFor

/*Select which offers to send*/
$k_1 = p_{pnew} * k$
$X_1 = \underset{\substack{C\subset P_{new} \\ |C|=k_1}}{\text{argmin}} \sum_{j=1}^{C} \min_{x \in X^t_{b\to a}} ||c_j - x||$

$k_2 = (1 - p_{pnew}) * k$
$X_2 = \underset{\substack{D\subset P \\ |D|=k_2}}{\text{argmin}} \sum_{j=1}^{D} \min_{x \in X^t_{b\to a}} ||d_j - x||$

$X^{t+1}_{a\to b} = X_1 \cup X_2$

Send $X^{t+1}_{a\to b}$

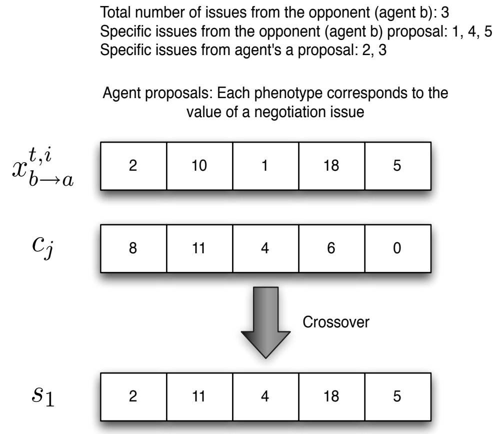

Figure 1: An example of a crossover operation

that contains new proposals discovered during the negotiation process (P_{new}). An example of a crossover operation can be observed in Figure 1.

A total of n_{mut} mutation operations are carried out for each generated child by crossover operations. This process generates new children that are also added to P_{new}. The mutation operator changes issue values randomly, according to a certain probability of mutating individual issues (p_{attr}). When p_{attr} is low, mutated proposals are close to the original proposal. This is the desired behaviour in this work. The operator is applied n_{mut} times to each child that is produced by crossover operations and to the original proposals from the opponent.

Note that no offer from P_{new} is discarded even although its utility is considered to be too low for the current instant. The reason for this, is that proposals that are not currently acceptable may be interesting in future negotiation rounds due to the concession strategy. Furthermore, since they have genetic material from the opponent's proposals, they are more likely to be accepted.

Select which offers to send.

Once genetic operators have been applied over received proposals, it is neccessary to send up to k different offers to the opponent. In order to send these offers, k proposals from the current iso-utility curve are sent. The negotiation strategy defines a proportion of p_{pnew} offers to come from P_{new}. The rest of the offers come from the result of the first GA (P).

The proposals selected from P_{new} are those that come from the current iso-utility curve that minimize the distance from the offers received from the opponent in the previous negotiation round. This selection may be formalized as:

$$\underset{\substack{C\subset P_{new}\\ |C|=p_{pnew}*k}}{\text{argmin}} \left(\sum_{j=1}^{C} \min_{x\in X^t_{b\to a}} ||c_j - x|| \right) \tag{6}$$

On the other hand, the proposals selected from P are also in the current iso-utility curve. The total number of proposals corresponds to a proportion that is equal to $1 - p_{pnew}$. Proposals that are the closest to the last offers received from the opponent are the ones that are to be finally selected. The selection strategy can be formalized as:

$$\underset{\substack{D\subset P\\ |D|=(1-p_{pnew})*k}}{\text{argmin}} \left(\sum_{j=1}^{D} \min_{x\in X^t_{b\to a}} ||d_j - x|| \right) \tag{7}$$

The parameter p_{pnew} determines how relevant are the new proposals explored during the negotiation process with respect to the older proposals. When $p_{pnew} = 0$, the strategy ignores the results that come from P_{new}. Consequently, only proposals that were discovered during the first GA are sent to the opponent. In this particular case, the strategy is equivalent to sending the proprosals from the approximated iso-utility curve that are closer to the last offers received. A problem that may arise in this particular case is the skipping of interesting proposals that were not discovered during the first genetic exploration. In contrast, when $p_{pnew} = 1$, the proposals discovered by genetic operations during the negotiation process are the only one taken into account. Problems that arise from this approach are related to the fact that the proposals that were discovered during the first exploration phase (which are usually larger in number than the ones discovered during the negotiation process) are ignored. Thus, agreement probability may be reduced considerably. Additionally, genetic operators do not guarantee anything about the utility of the new proprosals. These new proposals may only be introduced in later rounds. Intuitively, mixing both strategies ($p_{pnew} > 0 \ \wedge \ p_{pnew} < 1$) seems like a good approach. In any case, the parameter p_{pnew} is a strategy parameter to be adjusted.

3. EXPERIMENTS

The performance of the devised strategy is detailed in this section. First, the design of the experiments is described and then the results and a brief discussion are presented.

3.1 Experimental setting

It should be pointed out that the utility function used was the constraint-based model proposed in [5]. It has been acknowledged that there are mechanisms that work very efficiently in the constraint domain [13]. However, the goal of proposed strategy is to be general and not to depend on a specific utility function.

The aim of these experiments was to evaluate wheter or not the proposed strategy is capable of working in domains where it is not possible to completely explore agent preferences (i.e., negotiations with a large number of issues). More specifically, the goal was to determine wheter the use of genetic operators during the negotiation process helps to achieve better agreements. Different negotiation cases were randomly created for this purpose using the following settings:

- Number of issues $n_i = \{4,5,6\}$. The number of issues was limited so that theoretical results could be calculated.
- Integer issues. The domain for each issue was set to $[0, 9]$.
- $n_i{*}5$ uniformly distributed constraints per agent. For instance if n_i=4, there are 5 unary constraints, 5 bi-

nary constraints, 5 trinary constraints and 5 quaternary constraints.

- Utility for each n-ary constraint drawn randonmly from $[0, 100 * n]$. The utility is normalized to [0,1] for theoretical results.
- Constraint width for each issue uniformly drawn from $[2, 4]$.
- Agent deadline $d = 10$. Agents do not know their opponent's private deadlines.
- Number of proposals per round $k = 5$.
- Agent reservation utility $RU = 0$. Agents do not know their opponent's private reservation utilities.

For each number of issues, a total of 100 negotiation cases were generated with the above settings. Each case was repeated 30 times. The experiments were coded in C++ and run on a 4x2.83 Ghz Quad-Core Intel Core2 with 8Gb memory under Ubuntu 9.04.

Some of the measures used are based on theoretical results such as the Nash equilibrium and the Pareto frontier; others are based on practical aspects:

- The joint utility $(u_1 * u_2)$: It measures the quality of the agreement for both agents. Extreme results like $u_1 = 1, u_2 = 0$ are punished with a joint utility of zero.
- The euclidean distance to the closest Pareto frontier point
- The euclidean distance to the Nash equilibrium point
- The number of negotiation rounds

3.2 Results

The parameters for the first GA were set to $p_{cr} = 0.8$, $p_{dc} = 0.7$, and $n_{gen} = 100$. The number of input proposals for the negotiation process was set to $|P| = 8192$. The best results for the proposed strategy were found when $p_{pnew} = 0.7$, $M = 15$, $n_{cross} = 5$, $n_{mut} = 3$ and $p_{attr} = 0.3$. The results for $p_{pnew} = 0$ are also included in this section since they represent the strategy when no genetic operator is applied to received offers. The results showing the joint utility, euclidean distance to Nash equilibrium, euclidean distance to the closest Pareto optimal point, and the number of negotiation rounds can be found in Tables 1, 2, 3 and 4, respectively. These tables show two different types of results. First, they show the confidence interval for each measure taking into account every case and every experiment repetition. Second, they show the number of cases where the use of genetic operators during the negotiation process ($p_{pnew} = 0.7$) was statistically worse (L) than ignoring genetic operators ($p_{pnew} = 0$), the number of cases where there were no statistical differences (D), and the number of cases where $p_{pnew} = 0.7$ performed better (W).

The four tables present similar results with respect to the performance when using genetic operators during the negotiation process. The joint utility was higher when $p_{pnew} = 0.7$, which results in fair agreements. The difference between using genetic operators or not using them is small when $n_i = 4$. However, as the number of issues increases, the difference

Joint Utility $(u_a * u_b)$

N.issues	$p_{pnew} = 0.7$	$p_{pnew} = 0$	L	D	W
4	[0.59-0.61]	[0.53-0.55]	9	37	54
5	[0.55-0.56]	[0.44-0.45]	5	26	69
6	[0.51-0.52]	[0.37-0.38]	0	16	84

Table 1: Joint Utility: The left side of the table shows the confidence intervals of joint utility for every case and repetition. The right side shows the number of cases where the genetic-aided strategy($p_{pnew} = 0.7$) obtained statistically worse results (L), statistically equivalent results(D), and statistically better results(W) than the case where $p_{pnew} = 0$

Euclidean distance to Nash Equilibrium

N.issues	$p_{pnew} = 0.7$	$p_{pnew} = 0$	L	D	W
4	[0.12-0.13]	[0.18-0.19]	9	44	47
5	[0.13-0.14]	[0.23-0.24]	9	25	66
6	[0.16-0.17]	[0.30-0.31]	0	27	73

Table 2: Nash Distance: The left side of the table shows the confidence intervals of the average Nash distance for every case and repetition. The right side shows the number of cases where the genetic-aided strategy($p_{pnew} = 0.7$) obtained statistically worst results (L), statistically equivalent results(D) and statistically better results(W) than the case where $p_{pnew} = 0$

between the two methods becomes greater. It can be observed that there is a tendency for the performance of the non-genetic method ($p_{pnew} = 0$) to be greatly degraded as the number of issues gets larger. Nevertheless, this decrease is greatly reduced when genetic operators are applied. This is specially interesting in large-issue domains, which is the goal of this work and the case of real-world domains. Additionally, the number of cases where $p_{pnew} = 0.7$ performed worse was just 9, which corresponds to 9% of the cases. For the rest of the cases the method was equal (37%) or better(54%).

Something similar can be observed in the case of distance to Nash equilibrium and distance to the closest Pareto optimal point. When the number of issues was small, both methods obtained results that were very close to the Pareto frontier. This also was the case, to a lesser degree, for the distance to Nash equilibrium. However, the distance achieved by $p_{pnew} = 0$ almost doubled when $n_i = 6$; whereas the distance only slightly increased when $p_{pnew} = 0.7$. Similar results to the case of the joint utility can be observed when each individual case is analyzed.

Another interesting result is the one obtained in the number of rounds, where the use of genetic operators helped to achieve faster agreements. This factor is very important in real negotiation domains where time is of the utmost importance (i.e., dynamic markets, real-time domains, and so forth). Similarly, the difference between the two approaches was increased as the number of issues got larger.

In summary, these experiments have shown that the use of genetic operators over received and one's own proposals leads to better results in cases where it is not possi-

Euclidean distance to closer Pareto Optimal point

N.issues	$p_{pnew} = 0.7$	$p_{pnew} = 0$	L	D	W
4	[0.034-0.039]	[0.090-0.097]	10	37	54
5	[0.040-0.045]	[0.129-0.137]	3	33	64
6	[0.049-0.053]	[0.180-0.189]	0	18	82

Table 3: Pareto Distance: The left side of the table shows the confidence intervals of the average Pareto distance for every case and repetition. The right side shows the number of cases for where the genetic-aided strategy($p_{pnew} = 0.7$) obtained statistically worst results (L), statistically equivalent results(D) and statistically better results(W) than the case where $p_{pnew} = 0$

Negotiation rounds

N.issues	$p_{pnew} = 0.7$	$p_{pnew} = 0$	L	D	W
4	[3.79-3.88]	[4.44-4.55]	2	43	55
5	[4.12-4.21]	[5.21-5.32]	4	24	72
6	[4.27-4.34]	[5.72-5.83]	0	11	89

Table 4: Negotiation rounds: The left side of the table shows confidence intervals of the average negotiation rounds for every case and repetition. The right side shows the number of cases where the genetic-aided strategy($p_{pnew} = 0.7$) obtained statistically worst results (L), statistically equivalent results(D) and statistically better results(W) than the case where $p_{pnew} = 0$

ble to completely explore agent preferences. Even though when the number of issues is small, the non-genetic-based approach achieves results that are similar to the genetic approach, the genetic approach is still slightly better. However, as the number of issues increases, so do the differences between the two approaches. The performance of the non-genetic approach is greatly reduced, whereas the genetic approach only suffers from slight performance reductions. This demonstrates the appropriateness of genetic operators in reducing the impact of a large number of issues on the negotiation process. Moreover, the genetic-based method is capable of obtaining faster agreements, which is very important in real-world domains.

4. DISCUSSION AND RELATED WORK

In the last few years, most of the work in automated negotiation has focused on offering solutions for the case of imperfect knowledge and bounded computational resources. The use of *heuristics* is neccessary to provide a solution to problems of this type. This present work can be classified within this same category of solutions.

Faratin et al. [2] presented a negotiation strategy for bilateral bargaining that is focused on achieving *win-win* situations by means of trade-off. The heuristic applied to perform trade-off is similar to the one employed in this present work. Given an agent's current utility, the offer from the iso-utility curve that is most similar to the last offer received from the opponent is sent. The idea behind this heuristic is that, since the proposed offer is the most similar to the last offer received from the opponent, it is more likely to be satisfactory for both participants. A fuzzy similarity criteria is employed to compare offers. Nevertheless, the use of fuzzy similarity requires some knowledge of opponent preferences. The application of criteria of this kind is complicated in complex utility functions due to the inter-dependencies among the different issues. In this present work, we employ the euclidean distance, which does not require any knowledge about the opponent and which is independent of the inter-dependencies among issues.

The seminal work of Krovi et al. [9] opened the path for GA's in automated negotiation. Krovi et al. proposed a GA for bilateral negotiations that was performed each time a negotiation round ended. The population of chromosomes was randomly initialized with 90 random offers and 10 heuristic offers (the last offer from the opponent and the nine best offers from the previous round). The idea behind using GA's is that resulting the offers have good characteristics for both agents. However, 60 generations were needed each round in order to obtain the next proposal. Moreover, the strategy was devised for linear utility functions that had very few negotiation issues. The performance of this method becomes uncertain when there is a large number of issues or complex utility functions are used. Our work also employs GA's to obtain new proposals, but it is capable of providing solutions for domains with complex utility functions and domains where the number of issues is large.

Robu et al. [16] presented a non-mediated bilateral negotiation strategy for agents in electronic commerce. Agent utility functions are based on special graphical models called utility graphs. One of the agents, the seller, is responsible for finding agreements that are satisfactory for both parties. In order to do that, the seller models the buyer by means of utility graphs and tries to learn the buyer's preferences. However, utility graphs are only designed for binary issues. Our work differs in that it is capable of working with general complex utility functions and is also capable of working issue domains that are not necessarily binary.

In Lai et al. [10], a powerful heuristic for bilateral bargaining with general utility functions is presented. The negotiation protocol is based on the Rubinstein alternating protocol [15], but each agent is allowed to send up to k different offers in each round. The offer with highest utility is chosen from the k offers received from the opponent in the last round. The iso-utility curve of the agent is calculated. Then the offer from the curve that is the most similar to the one chosen by the agent from the offers made by the opponent is selected. This offer from the iso-utility curve becomes a seed from which *k-1* offers in the neighbourhood are randomly generated. The selected offer from the curve and the *k-1* generated offers are sent back to the opponent. Again, the general ideal behind this heuristic is that since the offers are similar to one of the last offers received from the opponent, they are more likely to be satisfactory for both parties. Even though this strategy does obtain results that are very close to Pareto optimality, the tests were performed with few negotiation issues. This strategy also assumes that it is possible to calculate the whole iso-utility curve, which requires a complete exploration of agent preferences. Our work is inspired on Lai et al. proposal [10]. Nevertheless, our work assumes that the number of issues may be large and that agent preferences may frequently vary. Consequently, it is too expensive to completely explore agent preferences and to calculate the iso-utility curve in a complete way. Therefore, agent preferences are explored by means of a GA in order to discover good utility proprosals. During the negotiation process the exploration of preferences continues using genetic operators on one's own good proposals and the proposals received from the opponent. The idea is to discover new proposals that are satisfactory for both parties. Our results show that when the number of issues is large, the negative impact on performance is lower for genetic operators than for the non-genetic method.

5. CONCLUSIONS AND FUTURE WORK

This paper presents a multi-issue bilateral bargaining strategy. It employs a non-mediated alternating protocol based on the exchange of k proposals each round. The proposals are sent from the current iso-utility curve; more specifically, the ones that are closest to the last offers sent by the opponent. The strategy is designed for scenarios with a large number of issues. Consequently, preferences cannot be explored completely, and iso-utility curves cannot be calculated properly. First, a genetic algorithm that is based on niching mechanisms is applied before the negotiation process in order to obtain good quality proposals. During the negotiation process, genetic operators are applied over the opponent's offers and the agent's own proposals. The goal is to discover new proposals that are satisfactory for both parties.

The results show that the use of genetic operators introduced during the negotiation process achieves better results in joint utility, distance to Nash equilibrium, distance to closest Pareto optimal point, and number of negotiation rounds. Moreover, as the number of issues becomes larger, genetic operators considerably reduce the negative effect on these measures. This makes the designed strategy more appropriate for scenarios with a large number of issues.

Future work in this direction includes studying the effect of changing preferences during the negotiation process, (i.e., when the strategy is integrated with an argumentation mechanism), and introducing different agent behaviours (more self-interested, more cooperative, etc) by means of some modifications in genetic and selection operators.

6. REFERENCES

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

[2] P. Faratin, C. Sierra, and N. R. Jennings. Using similarity criteria to make negotiation trade-offs. In *4th International Conference on Multi-Agent Systems (ICMAS-2000)*, pages 119–126, 2000.

[3] S. Fatima, M. Wooldridge, and N. R. Jennings. Optimal negotiation of multiple issues in incomplete information settings. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1080–1087, Washington, DC, USA, 2004. IEEE Computer Society.

[4] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[5] H. Hattori, M. Klein, and T. Ito. Using iterative narrowing to enable multi-party negotiations with multiple interdependent issues. In *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–3, New York, NY, USA, 2007. ACM.

[6] J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, 1975.

[7] C. Jonker and V. Robu. Automated multi-attribute negotiation with efficient use of incomplete preference information. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1054–1061, Washington, DC, USA, 2004. IEEE Computer Society.

[8] M. Klein, P. Faratin, H. Sayama, and Y. Bar-Yam. Protocols for negotiating complex contracts. *IEEE Intelligent Systems*, 18(6):32–38, 2003.

[9] R. Krovi, A. C. Graesser, and W. E. Pracht. Agent behaviors in virtual negotiation environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 29(1):15–25, 1999.

[10] G. Lai, K. Sycara, and C. Li. A decentralized model for automated multi-attribute negotiations with incomplete information and general utility functions. *Multiagent Grid Syst.*, 4(1):45–65, 2008.

[11] M. Luck and P. McBurney. Computing as Interaction: Agent and Agreement Technologies. In *Proc. IEEE Conf. Distributed Human-Machine Systems*, pages 1–6, 2008.

[12] S. W. Mahfoud. *Niching methods for genetic algorithms.* PhD thesis, Champaign, IL, USA, 1995.

[13] I. Marsa-Maestre, M. A. Lopez-Carmona, J. R. Velasco, and E. de la Hoz. Effective bidding and deal identification for negotiations in highly nonlinear scenarios. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 1057–1064, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[14] O. J. Mengshoel and D. E. Goldberg. The crowding approach to niching in genetic algorithms. *Evol. Comput.*, 16(3):315–354, 2008.

[15] M. Osborne and A. Rubinstein. *Bargaining and Markets.* The Academic Press, 1990.

[16] V. Robu, D. J. A. Somefun, and J. A. La Poutré. Modeling complex multi-issue negotiations using utility graphs. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 280–287, New York, NY, USA, 2005. ACM.