Improving Multi-agent Negotiations Using Multi-Objective PSO Algorithm

Ahmad Esmaeili and Nasser Mozayani

Department of Computer Engineering, Iran University of Science & Technology, Tehran, Iran
esmaeili@comp.iust.ac.ir, mozayani@iust.ac.ir

Abstract. Negotiation over limited resources, as a way for the agents to reach agreement, is one of the significant topics in Multi-Agent Systems (MASs). Most of the models proposed for negotiation suffer from different limitations in the number of the negotiation parties and issues as well as some constraining assumptions such as availability of unlimited computational resources and complete information about the participants. In this paper we make an attempt to ease the limitations specified above by means of a distributive agent based mechanism underpinned by Multi-Objective Swarm Optimization (MOPSO), as a fast and effective learning technique to handle the complexity and dynamics of the real-world negotiations. The experimental results of the proposed method reveal its effectiveness and high performance in presence of limited computational resources and tough deadlines.

Keywords: Negotiation, Multi-Agent Systems, Multi-Objective Particle Swarm Optimization, PSO.

1 Introduction

Negotiation is the process in which a group of human or software agents are communicating to reach an agreement accepted by all of the participants over the allocation of a specific resource [1]. In other words, the agents negotiating over some issues are trying to maximize their personal payoffs while ensuring that an agreement is reached [2]. This topic is so wide in significance and application that we might get overwhelmed by the numerous researches in different branches like Distributed Artificial Intelligence (DAI) [1, 3], Social Psychology [4], Game Theory [5, 6] and electronic commerce [7, 8].

In most of the real-world negotiations, there is an extensive negotiation state space which causes nonlinear negotiation dynamics. Consequently, sub-optimal rather than optimal solutions are often reached. Although normative game-theoretic models [6] can be used to theoretically analyze the optimal results (e.g. Nash equilibrium) under restricted situations such as bi-lateral negotiations, they often fail to provide a line of actions for the parties to follow in order to reach the desired optimal result in real-world negotiations. On the other hand, the classic methods assume that information about all of the parties is available, while this cannot be true in real cases. This inevitable lack of enough information in real situations has been tried to be handled in newer

P. Jędrzejowicz et al. (Eds.): KES-AMSTA 2010, Part I, LNAI 6070, pp. 92–101, 2010.

models by means of considering prior probability distribution [9]. However finding a proper value for these probability distributions raises another problem.

In addition to the theoretical works, there are a number of evolutionary approaches reported in literature. For instance, in [10] the authors has tried to propose a GA based mechanism to model the dynamic concession matching behavior arising in bi-lateral negotiation situations. The drawback of this approach is that, the utility function of the opponent agent should be estimated and available in order to compute the fitness function for every chromosome. Rubenstein-Montano et.al. proposed a similar mechanism but for multi-party and multi-issue negotiations [2]. In [11] a chromosome represents a negotiation rule rather than an offer and its fitness is computed in terms of the number of times it has contributed to reach an agreement.

In this paper we propose an agent-based multi-party multi-issue negotiation mechanism, underpinned by fast, efficient PSO algorithm which is faster than GA. In our approach, in contrary to the ones mentioned before, no constraining assumption is made about the number of the parties, the availability of the information about negotiation space, and ignoring time pressure as a very important factor in real-world situations.

The rest of this paper is organized as follows: in section two we review the standard particle swarm optimization algorithm. We will see how this algorithm is applied for multi-objective problems. In section three, a basic negotiation mechanism is presented. This mechanism is used to compare with our work. Sections four presents our work in details. This work is evaluated in section five through a simple experiment. Finally section six presents the conclusion.

2 Particle Swarm Optimization

Particle swarm optimization (PSO), introduced by Kennedy [12], mimics the behavior of a swarm of insects or a school of fish. If one of the particles discovers a good path to food the rest of the swarm will be able to follow instantly even if they are far away in the swarm. Swarm behavior is modeled by particles in multidimensional space that have two characteristics: a position and a velocity. More technically we follow these steps: the swarm of particles at first is initialized with a population of random candidate solutions. They move iteratively through the *d*-dimension problem space to search the new solutions, where the fitness, *f*, can be calculated as the certain quality measure. Each particle has a position represented by a position-vector x_i (i is the index of the particle), and a velocity represented by a velocity-vector v_i. Each particle remembers its own best position so far in a vector $P_i = (p_{i1}, p_{i2}, \ldots, p_{id})$, and its *j-th* dimensional value is p_{ij}. The best position-vector among the swarm so far (global best) is then stored in a vector P_g, and its *j-th* dimensional value is p_{gj}. During the iteration time *t*, the update of the velocity from the previous velocity to the new one is determined by Equation 1. The new position is then determined by the sum of the previous position and the new velocity by Equation 2.

$$v_{ij}(t+1) = w(t)v_{ij}(t) + c_1 r_1 (p_{ij}(t) - x_{ij}(t)) + c_2 r_2 (p_{gj}(t) - x_{ij}(t))\,. \tag{1}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1)\,. \tag{2}$$

where *w(t)* is called as the inertia factor; r_1 and r_2 are the random numbers in the interval [0, 1]; c_1 and c_2 are positive constants and are called coefficient of the self-recognition component and social component respectively.

Without loss of generality, a Multi-Objective Optimization problem is the problem of simultaneously minimizing K objectives $f_k(\vec{\mathbf{x}}), k = 1,2,...,K$, of a vector $\vec{\mathbf{x}}$ in the feasible region Ω. That is,

$$\underset{\vec{\mathbf{x}} \in \Omega}{\text{minimize}} \; \vec{\mathbf{f}}(\vec{\mathbf{x}}) = \left[f_1(\vec{\mathbf{x}}), f_2(\vec{\mathbf{x}}), ..., f_K(\vec{\mathbf{x}}) \right]^T . \tag{3}$$

where $\vec{\mathbf{x}} = [x_1, x_2, ..., x_D]^T$ is a D-dimensional vector and $f_k(\vec{\mathbf{x}})$ (k = 1, 2, .., K) are linear or nonlinear functions. A vector $\vec{\mathbf{u}} = (u_1, u_2, ..., u_D)$ is said to strongly dominate $\vec{\mathbf{v}} = (v_1, v_2, ..., v_D)$ (denoted by $\vec{\mathbf{u}} \prec \vec{\mathbf{v}}$) if and only if $\forall i \in \{1,2,..,K\}, f_i(\vec{\mathbf{u}}) < f_i(\vec{\mathbf{v}})$. And a decision vector $\vec{\mathbf{u}}$ weakly dominates $\vec{\mathbf{v}}$ (denoted by $\vec{\mathbf{u}} \preceq \vec{\mathbf{v}}$) if and only if $\forall i \in \{1,2,..,K\}, f_i(\vec{\mathbf{u}}) \le f_i(\vec{\mathbf{v}})$ and $\exists i \in \{1,2,..,K\}, f_i(\vec{\mathbf{u}}) < f_i(\vec{\mathbf{v}})$.

It is obvious that the solutions dominated by other ones are not desirable. A set of decision vectors is said to be a non-dominated set if no member of the set is dominated by any other member. The true Pareto-optimal front is the non-dominated set of solutions which are not dominated by any feasible solution. One way to solving a multi-objective optimization problem is to approximate the Pareto-optimal front by the non-dominated solutions generating from the solution algorithm.

The difficulty in extending the PSO to be used in multi-objective optimization problems is how to select a global guide for each particle. Since there is not a single optimum in multi-objective optimization, the non-dominated solutions found by Multi-Objective PSO (MOPSO) so far are all stored in an archive. Each particle can randomly select a non-dominated solution from the archive as the global guide of its next flight. Although this selection method is simple, it can promote convergence [13].

3 A Simple Negotiation Mechanism

In this section we present a basic negotiation model which is employed by our approach. This simple model guarantees Pareto Optimum and is used in section five to evaluate the effectiveness of our work. A solution is Pareto optimal if it is impossible to find another solution such that at least one agent will be better off but no agent will be worse off [14]. The definition of this mechanism has adopted from [15].

In the simple model the negotiation space is represented by $Neg = \langle P, A, D, U, T \rangle$. In this tuple, P represents a finite set of participants, A represents a set of attributes (issues) used in negotiation and understood by all of the agents. An attribute domain is specified by D_{a_i} where $D_{a_i} \in D$ and $a_i \in A$. U denotes the set of utility functions in which the utility of agent $p \in P$ is denoted by $U_p^o \in U$ and defined by: $U_p^o : D_{a_1} \times D_{a_1} \times ... \times D_{a_1} \mapsto [0,1]$. Finally, the deadline for every agent p is represented by $t_p^d \in T$. In this model, it is assumed that information about *P, A, D* is exchanged among the negotiation participants during the *ontology sharing* stage before negotiation actually takes place. A *multi-lateral* negotiation situation can be modeled as many one-to-one *bi-lateral* negotiations where an agent *p* maintains a separate negotiation dialog with each opponent. In a negotiation round, the agent will make an offer to each of its opponent in turn, and consider the most desirable counter-offer from among the set of incoming offers evaluated according to its own payoff function U_p^o. An offer $\vec{o} = \langle d_{a_1}, d_{a_2}, ..., d_{a_n} \rangle$ is a tuple of attribute values (intervals) pertaining to a finite set of attributes $A = \{a_1, a_2, ..., a_n\}$. Each attribute a_i takes its value from the corresponding domain D_{ai}. The agent *p*'s valuation function for each attribute $a \in A$, and each attribute value $d_a \in D_a$, are defined by: $U_p^A : A \mapsto [0,1]$ and $U_p^{D_a} : D_a \mapsto [0,1]$ respectively. The valuations of attributes are assumed normalized, i.e. $\sum_{a\in A} U_p^A(a) = 1$. One common way to measure an agent's utility function for an offer *o* is as Equation 4:

$$U_p^o(o) = \sum_{a\in A} \left(U_p^A(a) \times U_p^{D_a}(d_a) \right). \tag{4}$$

If an agent's initial proposal is rejected by its opponent, it concedes with the least utility decrement. Agents keep a record of the offers they have made before and in the current round in set O'_p and any round an alternative offer with concession can be determined based on Equation 5.

$$\begin{aligned} &\exists o_{counter} \in \{O_p - O'_p\}, \\ &\qquad \forall o_x \in \{O_p - O'_p\} : [o_x \preceq_p o_{counter}]. \end{aligned} \tag{5}$$

It should be noted that here the relation $\preceq_p$ defines an ordering over the set of feasible offers O_p induced by an agent's utility function.

Now let's see how the incoming offers are processed in this model. When offer *o* is received from an opponent, it is evaluated to see if satisfy all of its constraints. This is carried out by computing an equivalent offer, $o_\approx$, as an interpretation about the opponent's proposal *o*. The offer $o_\approx$ is equivalent to *o* if and only if every attribute interval of $o_\approx$ intersects each corresponding attribute interval in *o*. Once it is computed, it decides whether to accept it according to the following criteria:

1. If $\forall o_x \in O_p : o_x \preceq_p o_\approx$ since the offer *o* produces the maximal payoff.
2. If $o_\approx \in O'_p$ is true. That is, the equivalent offer $o_\approx$ is one of the previously made offers or the one to be proposed in the current round.

It has been proved that following the mechanism specified above, Pareto optimum is always guaranteed in case it exists [15]. However, it should be noted that maximal joint payoff of all of the agents is not necessarily guaranteed.

4 Negotiation Based on Multi-Objective Particle Swarm Optimization

In this section we describe our approach to optimizing negotiations using the MOPSO algorithm described in section 2. The decision making model of this adaptive approach is based on the basic idea that, agents tend to maximize their own payoffs while ensuring that an agreement is reached [16]. The general process of the proposed method is as follows: in every multi-lateral negotiation, each agent identifies the best offer of the incoming offers and an offer with maximal payoff according to its utility function. In rest of this section these two offers are denoted by $o_{counter}$ and o_{max}, respectively. Then, a population of particles representing a subset of feasible offers is created and the distance from these two offers is measured as part of the fitness function. Using a simple encoding for the particles as presented further on, this distance can be easily computed using the general methods like weighted Euclidean distance.

4.1 Particle Encoding

As mentioned before, for each agent a population of particles is used to represent the feasible offers. Each particle is composed of as many fields as the number of negotiation issues. This number is fixed among all of the agents, and the *i*th-negotiation issue is represented by Neg_att_i (Fig. 1).

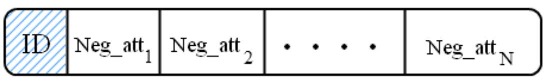

Fig. 1. Architecture of a particle

As it can be seen, each particle also consists of an extra field, namely *ID*, which is used to be able to keep track of the particles.

4.2 Fitness Function

The fitness function should be defined in such a way that the most profitable offer for an agent against the received counter-offer be the one with the highest fitness value. Ideally thinking, the fitness function should reflect the joint payoff of the offer represented by the particle it is applied for; however, in real-world situation the utility function of the opponents are not available to enable us to compute the joint pay off. Therefore, we must devise a method to approximate this ideal function. In the fitness function we propose here, we take three components into consideration. The first component takes a value in range [0, 1] and reflects the quality of the current offer to increase the agent's personal payoff. The closer it is to 1, the more it increases the agent's individual payoff. The second component deals with the concession to the latest offer received from the opponent. This component can be measured in terms of the distance between current offer and the one received from the opponent. The less this distance is, the more the agent concedes using this offer. To provide a balance between these two components, coefficient $SI \in [0,1]$ is introduced. This coefficient specifies the degree of the self-interestedness of the agent (It is considered fully self-interested if $SI = 1$). Finally in the third component, we consider the deadline in terms of time pressure (*Time*), which is applied as a coefficient to the previous two components. In this paper we use a simplified version of the polynomial time-dependant decision function in [9] to compute and apply this time pressure (Equation 6):

$$Time(t) = 1 - \left(\min\left(t, t_p^d\right) / t_p^d\right)^{1/\beta}. \tag{6}$$

In this function, *t* denotes the absolute time or the number of negotiation rounds and as defined before in section 3, t_p^d denotes the deadline for agent *p*. This deadline can also be specified either in absolute time or the maximum number of negotiation rounds that is allowed for agent *p*. Furthermore, β specifies the agent's attitude toward concession. This factor is defined by the user before the negotiation starts and may take positive non-zero values. The value in range (0, 1) specifies that the agent does not easily give ground during negotiation. An agent that shows such a behavior is called a *Boulware Agent* according to [17]. On the other hand, in case the value of β is set to be $\beta > 1$, it means that the agent is willing to concede (called *Conceder Agent*). If the agent is neutral to concession, β is set to 1.

Considering all of the details described above, the fitness function for MOPSO is defined as Equation 7:

$$fitness(o) = \omega \times \left(\frac{Utility(o)}{\max\_Utility} \right) + (1-\omega) \times \left(1 - \frac{\sqrt{\sum_{i=1}^{|A|} v_i (d_i^o - d_i^{o_{counter}})^2}}{\gamma} \right). \tag{7}$$

where $\omega = SI \times Time(t)$ is a controlling parameter specifying the relative significance of optimizing the agent's own utility or reaching an agreement; *Utility(o)* and *max_Utility* are the utility of the current offer specified by the current particle and the maximum payoff can be achieved, respectively. These two values can be easily determined using the agent's own utility function; $|A|$ is the number of negotiation attributes or issues; v_i is the valuation of the agent for the *i*th-attribute and is defined by the user; d_i^x denotes the *i*th-attribute value in offer *x*; *o* and $o_{counter}$ are respectively the offers represented by the current particle and the latest one received from the opponent; and finally γ is the maximum distance in the negotiation space. In computing the distance in the second part of the fitness function, we have linearly scaled the attribute values to take on values from the unit interval [0, 1].

5 Experimental Results

In order to evaluate our proposed mechanism, we created a multi-lateral negotiation situation composed of 3 Buyer agents and 3 Seller agents. These agents are negotiating over a product that is described by five attributes. Each of these attributes is of discrete type and takes the values form the set {-2, -1, 0, 1, 2}. The deadline of the negotiation is set to 200 rounds. In this experiment the properties of negotiations are set in the manner that an agreement zone surely exists.

In the beginning of each negotiation round, an agent runs its MOPSO algorithm to produce an offer. In this phase, since no offer has been received from any agent, the second part of the fitness function is set to be zero. Upon receiving the counter offers from the opponents, each agent chooses the best offer according to its own utility function (in case more than one offers are candidate for being the best offer, one is selected randomly). Now again MOPSO algorithm is run by that agent to produce a new offer in response. Typical two agents continue negotiating and exchanging messages until an agreement is reached or the pre-specified negotiation deadline is passed.

The configuration of the PSO algorithm parameters for this experiment is adjusted as Table 1.

Table 1. PSO Parameters

Parameters	**Values**
Swarm Size	60
Number of Iterations	1000
Acceleration Coefficients (c_1, c_2)	2
Inertia weight	0.9 to 0.4
V_{max}	0.5

We have conducted the complete negotiation process ten times, each on various configurations but with the same set of agents. In this experiment, the values of the parameters previously specified as part of the fitness function are initialized as Table 2.

Table 2. Fitness Parameters

Parameter	**Value**
SI (Buyer)	0.7
SI (Seller)	0.4
β	1

The same configuration (number of agents and their utilities) has also been used in the experiment to evaluate the basic negotiation mechanism described in section two. We have examined our work in various ways. At first we have compared our work with the Pareto optimal mechanism described in section 2. This comparison has been conducted on base of average payoff of the agents during the negotiation process and average length of time that it took for the agents to reach an agreement. The results are presented in Table 3.

Table 3. Average payoff and negotiation time

	(MOPSO)		Pareto Opt.			
Agent	**Avg. Payoff**	**Avg. Time**	**Avg. Payoff**	**Avg. Time**	**Δ payoff**	**Δ time**
B1	0.62	155.6	0.08	198.3	+0.54	-42.7
B2	0.63	153.1	0.1	199.8	+0.53	-46.7
B3	0.56	159.2	0.04	191.5	+0.52	-32.3
S1	0.43	162.8	0.04	199.1	+0.39	-36.3
S2	0.47	157.8	0.07	189.4	+0.40	-31.6
S3	0.42	162.9	0.08	199.8	+0.34	-36.9

Carefully observing this table, one can easily find out that the MOPSO-based negotiation mechanism surpasses the Pareto optimal one both in effectiveness (regarding the average payoffs) and efficiency (regarding average negotiation time). This success apparently is because the Pareto optimal approach is not able to find a solution in just 200 rounds. However, in our MOPSO-based approach, the agents succeeded to reach agreement because of being sensitive to the deadlines. Although the proposed approach does not guarantee to find the optimal solution, it is much more desirable in real-world situations especially in presence of the deadline.

In order to exhibit the concession behavior of the agents during the negotiation time, we have depicted the utility of one of the agents (Seller1 Agent), in different negotiation rounds (see Fig. 2). It is clear that concession process in our proposed method is very adaptive and sensitive to the deadline imposed to the negotiation environment and the utility of the agent Seller1 in MOPSO-based mechanism decreases rapidly as the deadline is reached. Furthermore, to examine stability of the proposed technique, we ran the last complete experiment for 5 times, and we recorded the average of the average payoffs obtained in each iteration for every agent. This average vs. the number of iterations is depicted for every agent in Fig. 3.

As it can be seen, the averages of an agent's payoffs under different number of iterations of negotiations are very close to each other. This proves stability in the quality of the solutions that our proposed mechanism produces. Moreover, according to Fig. 3 we observe that the average payoff values gained by the Buyer agents are greater than the ones obtained by the Seller agents. This phenomenon can be justified by the *SI* values we set for these two kinds of agents. As discussed before, the more the value of this parameter is, the greater payoff the corresponding agent will gain in case an agreement is reached.

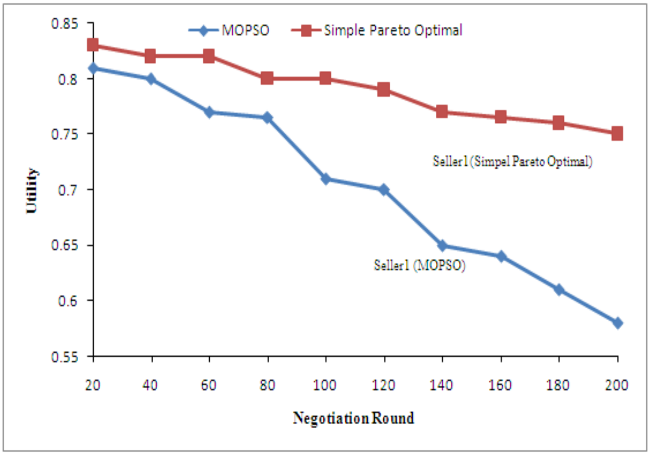

Fig. 2. Comparative concession behavior of MOPSP-based and Simple Pareto Optimal negotiation mechanisms

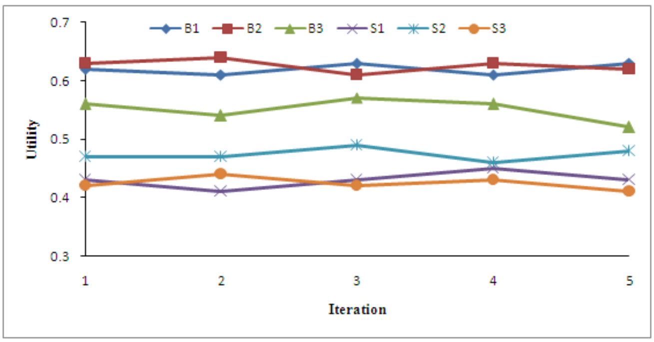

Fig. 3. The agents' average payoffs over different iterations

6 Conclusion

Real-world negotiations are constrained by various factors such as the unavailability of enough information, presence of deadline, great number of issues, and etc. These constraints cause most of the usual negotiation mechanisms to fail to give a reasonable solution. In this paper we tackled the problems by considering the negotiation as a multi-objective optimization problem and the decision making process of each agent was underpinned by the MOPSO algorithm to produce suitable offers. Our approach uses a very simple architecture and an effective fitness function, and can handle situations in which various numbers of agents are negotiating over multiple issues in presence of tough deadlines. We examined our proposed mechanism on a multi-party multi-issue negotiation environment with deadlines and compared it with a simple Pareto optimal system that we described in the paper. The experimental results and analysis presented in this paper proves the efficiency, effectiveness, stability and convergence of the proposed method in comparison with a Pareto optimal mechanism in real-world negotiations. Enhancement of the proposed method to take into account the factors such as competition and opportunity in multi-lateral negotiations is going to be considered as future work.

References

1. Sycara, K.: Multi-agent compromise via negotiation. In: Gasser, L., Huhns, M. (eds.) Distributed Artificial Intelligence II, pp. 119–139. Morgan Kaufmann, San Francisco (1989)
2. Rubenstein-Montano, B., Malaga, R.A.: A Weighted Sum Genetic Algorithm to Support Multiple-Party Multi-Objective Negotiations. IEEE Transactions on Evolutionary Computation 6(4), 366–377 (2002)
3. Kraus, S.: Negotiation and cooperation in multi-agent environments. Artificial Intelligence 94(1-2), 79–97 (1997)
4. Pruitt, D.: Negotiation Behaviour. Academic Press, London (1981)
5. Rubinstein: Perfect equilibrium in a bargaining model. Econometrica 50(1), 97–109 (1982)
6. von Neumann, J., Morgenstern, O.: The Theory of Games and Economic Behaviour. Princeton University Press, Princeton (1994)
7. Fatima, S., Wooldridge, M., Jennings, N.R.: Comparing Equilibria for Game-Theoretic and Evolutionary Bargaining Models. In: Proceedings of the International Workshop on Agent-Mediated Electronic Commerce V, Melbourne, Australia, pp. 70–77 (2003)
8. He, M., Jennings, N.R., Leung, H.: On agent-mediated electronic commerce. IEEE Trans. on Knowledge and Data Engineering 15(4), 985–1003 (2003)
9. Harsanyi, J., Selten, R.: A generalised nash solution for two-person bargaining games with incomplete information. Management Sciences 18(5), 80–106 (1972)
10. Krovi, R., Graesser, A., Pracht, W.: Agent behaviors in virtual negotiation environments. IEEE Transactions on Systems, Man, and Cybernetics 29(1), 15–25 (1999)
11. Matwin, S., Szapiro, T., Haigh, K.: Genetic algorithm approach to a negotiation support system. IEEE Transactions on Systems Man and Cybernetics 21(1), 102–114 (1991)
12. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. of IEEE Int. Conf. on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)
13. Alvarez-Benitez, J.E., Everson, R.M., Fieldsend, J.E.: A MOPSO algorithm based exclusively on Pareto dominance concepts. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 459–473. Springer, Heidelberg (2005)
14. Rosenschein, J., Zlotkin, G.: Task oriented domains. In: Rules of Encounter: Designing Conventions for Automated Negotiation among Computers, pp. 29–52. MIT Press, Cambridge (1994)
15. Barbuceanu, M., Lo, W.K.: Multi-attribute utility theoretic negotiation for electronic commerce. In: Dignum, F., Cortés, U. (eds.) AMEC 2000. LNCS (LNAI), vol. 2003, pp. 15–30. Springer, Heidelberg (2001)
16. Faratin, P., Sierra, C., Jennings, N.R.: Using similarity criteria to make issue trade-offs in automated negotiations. Artificial Intelligence 142(2), 205–237 (2002)
17. Raiffa, H.: The Art and Science of Negotiation. Harvard University Press, Cambridge (1982)