Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 597–603
ISBN 978-83-60810-22-4

Computing Equilibria for Constraint-based Negotiation Games with Interdependent Issues

Mihnea Scafeş
University of Craiova
Software Engineering Department
Bvd.Decebal 107, Craiova, 200440, Romania
scafes_mihnea@software.ucv.ro

Costin Bădică
University of Craiova
Software Engineering Department
Bvd.Decebal 107, Craiova, 200440, Romania
badica_costin@software.ucv.ro

***Abstract*—Negotiation with interdependent issues and nonlinear, non-monotonic utility functions is difficult because it is hard to efficiently explore the contract space. This paper presents a new result in automated negotiations with interdependent issues, complete information and time constraints. We consider that agents express their preferences using constraints defined as one interval per issue and that we represent their constraint sets as intersection graphs. We model negotiations as a bargaining game and we show that the equilibrium solution is one of the maximal cliques of the constraint graph. Consequently, we find that the problem of computing the equilibrium solution has polynomial-time complexity when the number of issues is fixed.**

I. Introduction

MOST of the research in negotiation has considered that the negotiation issues are independent, i.e. the value of one issue does not depend on the values of the other issues. In this case the utility functions are typically linear and monotonic. As the agents often have to make concessions to the other agents, while also maximizing their utility, they must determine contracts that, depending on a given situation, will either increase or decrease the utility values. If the utility functions are monotonic, it is computationally easy to decide for either increasing or decreasing the value of an issue in a potential contract in such a way that the value of the utility function will be increased or decreased. But, if there are multiple possible trade-offs between issues, computing the agreement is computationally more expensive.

Real-world applications may require negotiation with interdependent issues which are typically aggregated in a more complex way than simple linear utility functions. Examples can be given in areas as meeting scheduling [1], [2], cooperative design [3], and energy markets [4]. In these cases utility functions are nonlinear and non-monotonic, so searching the contract space for contracts that score above a given utility value requires an efficient exploration method because the number of contracts is exponential.

The interdependencies between issues can be represented in several ways: utility graphs [5], [6], interval constraints [7]–[9] and influence matrices [10]. Most of the approaches for finding the optimal outcome of the negotiation implement simulated annealing algorithms [6]–[10]. This approach has the disadvantage that agents may accept contracts of a lower utility value, rather than accepting only contracts of a higher utility value as with the hill-climbing method. Thus, an agent employing a simulated annealing strategy will get a lower outcome from a negotiation with an agent employing a hill-climbing strategy [10]. The simulated annealing approach was implemented using a mediator for computing contracts and most of the time it determines near-optimal solutions because computing an optimal contract is time expensive.

In our work we focus on bargaining with complete information, under time constraints, about interdependent issues. When modelling such negotiations, one of the most important things is to study the equilibrium solution. A general framework for finding the equilibrium solution for this class of problems has already been proposed in the literature [11]. The application of this framework produced a series of interesting results, under certain conditions regarding the negotiation configuration [11]–[13].

We build our work on the same framework for finding the equilibrium solution of our negotiation problem. By using a preference model suited for interdependent issues [8] and modelling negotiation as a bargaining game under time constraints, we define and solve a problem that leads us to the equilibrium solution. But probably the most important aspect is that we prove that this problem can be solved in polynomial time when the number of negotiation issues is fixed.

According to our literature review, there is no other work that gives the same important results in the context of bargaining. However, the preference model that we use here has been already considered in previous works [8], although in different negotiation settings. Here we are studying negotiation as bargaining game with complete information, differently from the mediated, auction-based mechanisms used in other works [8]. Moreover, our results can be successfully applied to other negotiation settings.

The paper is structured as follows. We start in Section II with an analysis of the preference model with interdependent issues. We briefly describe in Section III the negotiation protocol and the structuring of the offers. In Section IV we model negotiation as a bargaining game of alternating offers and we prove that the equilibrium solution can be computed in polynomial time. Section V provides an overview of related research on negotiation with interdependent issues. In Section VI we draw conclusions and we point to future work.

II. Preferences and utility

There is a set of m negotiation issues $\mathcal{X} = \{x_1, x_2, \dots x_m\}$. Issues can be assigned values only from their domain. There are m domains, $D_1, D_2, \dots D_m$. A *contract* is an m-dimensional variable defined over $D_1 \times D_2 \times \dots \times D_m$, i.e. a combination of issue values. The set of possible contracts is therefore $D_1 \times D_2 \times \dots \times D_m$.

Agents have preferences over the issues in the form of constraints.

Definition 1: Let $\mathcal{X} = \{x_1, x_2, \dots x_m\}$ be the set of issues under negotiation and $D_1, D_2, \dots D_m$ their domains. A constraint C is a boolean function defined over the set of possible contracts, i.e. $C : D_1 \times D_2 \times \dots \times D_m \to \{0, 1\}$. We say that an arbitrary constraint C is satisfied by a contract x when $C(x) = 1$.

A constraint restricts the domains of the issues to smaller sets of values and introduces a set of contracts $Dom(C) = \{x \in D_1 \times \dots \times D_m | C(x) = 1\}$. We refer to $Dom(C)$, i.e. the set of contracts x for which $C(x) = 1$, as the *domain of constraint* C.

But as agents usually have complex preferences that are modeled by combining several constraints, we need to know if these constraints are compatible.

Definition 2: A constraint set $\mathcal{C}$ is *consistent* if $\exists x \in D_1 \times D_2 \times \dots \times D_m$ such that $C(x) = 1, \forall C \in \mathcal{C}$.

A consistent constraint set defines a domain equal to the intersection of the individual domains defined by each constraint of the set. For example, if we have a constraint set $\mathcal{C} = \{C_1, C_2, C_3\}$, then the *domain of constraint set* $\mathcal{C}$ is $Dom(\mathcal{C}) = Dom(C_1) \cap Dom(C_2) \cap Dom(C_3)$. An inconsistent constraint set defines an empty domain that contains no contracts.

We can associate an intersection graph to the constraint set of an agent. We associate a vertex to each constraint. Two vertices are connected by an edge if the domains of their associated constraints intersect. If $\mathcal{C}$ is the set of constraints then the associated constraint graph is denoted by $G_{\mathcal{C}}$.

In what follows we will refer to this intersection graph as the constraint graph. If we assume that each agent is expressing his preferences using a constraint set (which might be consistent or not as a whole), he will have such a constraint graph.

Please note that a constraint graph, say $G = (V, E)$, is composed by intersecting m graphs $G_i = (V, E_i)$, $1 \le i \le m$, one for each issue, which share the same vertex set, but not the edges. A vertex in graph G_i corresponding to constraint C is assigned the domain of issue i which is part of constraint C. The constraint graph G has the same vertex set as G_i, $1 \le i \le m$. Two vertices in graph G are connected by an edge if the two vertices are connected in all graphs G_i, i.e. $(x, y) \in E$ if and only if $(x, y) \in E_i$, $\forall x, y \in V$, $\forall 1 \le i \le m$. We refer to graphs G_i as *issue graphs*.

The following proposition describes how a consistent constraint set looks like in the constraint graph.

Proposition 1: Any consistent constraint subset defines a clique in the constraint graph.

Proof: Let $\{C_1, C_2, ..., C_k\}$ be a consistent constraint subset. It follows that $Dom(C_1) \cap Dom(C_2) \cap ... \cap Dom(C_k)$ is non empty. Then for each $i \neq j$, $Dom(C_i) \cap Dom(C_j) \neq \emptyset$, so the constraint graph associated to the constraint set $\{C_1, C_2, ..., C_k\}$ is a clique. Therefore, a consistent constraint subset defines a subgraph of the constraint graph in which all the vertices are connected with each other, i.e. a clique. ■

The reverse is not always true, i.e. not every clique corresponds to a consistent constraint set. Note that as a consistent constraint set defines a non-empty domain, we can say that a clique in a constraint graph also defines a non-empty domain.

Agents negotiate about a set of issues, seeking to increase their outcomes. The following definition introduces the utility function that agents use to evaluate contracts. We assume that the agents use the preference model described earlier in this section. This model of utility functions is adopted from [8].

Definition 3: Let $\mathcal{C}_A$ be the constraint set of agent A (the set of all constraints used by agent A to express his preferences). Each constraint $C \in \mathcal{C}_A$ has an associated weight (a strictly positive real number) $\omega_C > 0$, which the agent uses to build a preference relation over constraints. These weights are normalized, i.e. $\sum_{C \in \mathcal{C}_A} \omega_C = 1$. The function $u_A : D_1 \times D_2 \times \dots \times D_m \to \mathbb{R}$, $u_A(x) = \sum_{C \in \mathcal{C}_A} \omega_C \times C(x)$ is the utility function of agent A.

The utility function corresponding to outcome x is the weighted sum of all constraint evaluations in x. Throughout the paper we assume that agents try to maximize their utility functions [14].

Note that the utility function for independent preferences is usually modeled as a weighted sum of issue values. However, even if it has a similar form (i.e. a weighted sum), the utility function from Definition 3 defines a complex aggregation of constraints and usually has a non-monotonic shape [8].

An agent must determine consistent combinations of constraints in order to find contracts that score above a particular utility threshold. Please note that not every value in $[0, 1]$ can be scored by the utility function, as the set of all possible values scored by the utility function is finite. Compared to the linear programming problems studied earlier for independent utilities [11], [13], exploration of the contract space becomes a combinatorial optimization problem.

For the rest of the paper we make a very important assumption, that drives the result of our work. We assume that a constraint defines a single interval (open or closed) on the real line for each negotiation issue. Example 1 illustrates a constraint set with such constraints. A constraint is therefore defined as a conjunction of interval memberships on the real line, with one interval per issue. Please note that in this case the issue graphs are interval graphs [15] and a constraint graph is composed by intersecting m interval graphs. Therefore, under this assumption, the reverse of Proposition 1 is true, i.e. each clique is a consistent constraint subset[1].

[1] This statement results from the fact that real intervals have the Helly property: if a set of real intervals is such that each two of them intersect then all of them have a non-empty intersection.

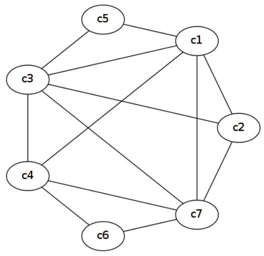

Fig. 1. Example of constraints. Weights are not illustrated, for simplicity. Edges connect vertices whose domains intersect

Example 1: Let's suppose that an agent has the following simple constraints over 2 issues denoted x, y, defined over $[0, 100]$: c1 ($x \in [37, 84]$), c2 ($x \in [79, 96]$), c3 ($x \in [41, 86] \wedge y \in [5, 78]$), c4 ($x \in [5, 51] \wedge y \in [33, 98]$), c5 ($x \in [61, 69] \wedge y \in [36, 40]$), c6 ($x \in [14, 25]$), c7 ($y \in [62, 69]$). When the interval for an issue is not specified, it is assumed that the domain of the issue is not restricted. The constraint set is depicted in Figure 1.

III. Protocol

We model negotiation as a bargaining game of alternating offers [16]. In this paper we study only bilateral negotiations, i.e. negotiations with 2 players denoted by A and B.

In the general case, an *offer* consists of a set of values for each issue. The set assigned to an issue must be included in the domain of the issue. An agreement is represented by the last offer that has been accepted, so it also consists of a set of values for each issue. We consider that agents are indifferent about the issue values chosen from these sets and an exact agreement (i.e. that consists of exactly one value from each of these sets for each issue) can be chosen randomly by one of the players.

Following our preference model described earlier, an offer contains a list of *offered constraints* with one interval per issue[2]. The offered constraints can be constraints from the constraint set of the agent or constraints formed from constraints in the constraint set by shrinking their intervals. Shrinking of an interval happens when an interval is intersected with another interval, as a result of combining constraints into consistent constraint sets. In other words, the constraints in the offers are composed of one interval per issue and these intervals represent domains of consistent constraint subsets (or cliques in the constraint graph) of the agents.

The agent receiving an offer can accept the offered constraints or he can accept only parts of offered constraints obtained by shrinking their domains (i.e. by shrinking one or several issue intervals). This operation makes sense as agents try to satisfy as many of their constraints as possible, and sometimes shrinking domains results in contracts that satisfy more constraints. In this case, shrinking happens when the agent makes combinations of constraints to determine consistent constraint sets and intersects their domains with the domains in the opponent's offer. In this way he can accept only a part of the offer that maximizes his utility.

Agents seek to maximize their private utilities, but it is required that the opponent's utility is not lost, i.e. Pareto agreements are preferred. An agreement is Pareto if there is no other agreement that is at least as good for both agents and strictly better for at least one of them.

[2]We are consistent with the assumption made earlier. In the most general case an offer can contain a more complex specification of a set of values per issue, for example a union of intervals, rather than a single interval. However, we do not deal with this case in this paper. Our results are based on the assumption that there is only a single interval assigned to one issue in a constraint, as in [8].

IV. Negotiation with complete information

Let A and B be two agents that negotiate in order to reach an agreement over their sets of preferences. They would like to allocate values to multiple issues, but the issues are constrained to subsets of their domains. The issues are interdependent, i.e. the subset of values that an issue is allowed to take depends on the subsets of values that the other issues can take. These interdependencies are modeled using multi-issue constraints, as described in Section II. Each agent has a constraint set containing possibly many such constraints. Agents A and B have constraint sets C_A and C_B respectively.

Agents have time constraints given as deadlines and discount factors similarly to [11]–[13]. We represent deadlines as time steps in negotiation. For simplification (but without losing the generality of the algorithm), we make the assumption that both agents have the same deadline, denoted with n. We assume that agents have no gain if the agreement is reached after the deadline and so agents wish to reach agreements before their deadlines. Moreover, we assume that the utility of each agent decreases as negotiation advances in time. We model this with the help of a discounted utility function that depends on two variables: the contract and the negotiation step. By taking into account deadlines and discount factors, the agents $i \in \{A, B\}$ have the discounted utility function:

$$U_i(x,t) = \begin{cases} u_i(x) \times \delta_i^{t-1} & \text{if } t \leq n \\ 0 & \text{if } t > n \end{cases}$$

with $\delta_A \in [0, 1)$, $\delta_B \in [0, 1)$ and $t \geq 1$. Again, to reduce the complexity of the representation (but not the generality), we assume that both agents have the same discount factor δ. Thus, the utility an agent gets from a contract in the first negotiation round is greater than the utility it gets from the same contract in subsequent negotiation rounds.

Please note that the negotiation game we study is not the *split-the-pie* game [17] previously studied for independent issues. There is no object that the participants want to split and that shrinks over time. Participants want to reach an agreement over a set of values (that are constant during the negotiation), not over shares of an object.

We consider that agents have complete knowledge about the deadline and the discount factor and they know each other's preferences. Our negotiation model builds on existing negotiation models with complete information [11]. We adapt existing equilibrium strategies for negotiation with complete information, under time constraints, with multiple issues and monotonic utility functions [11] to negotiation with complete information, under time constraints, with multiple interdependent issues and non-linear, non-monotonic utility functions. We show that under specific circumstances (the preferences are expressed using constraints that restrict issue values to intervals and the utility functions are weighted sums of constraint evaluations) the equilibrium solution is not as hard to compute as it was recently considered. Although the same preference model under the same circumstances had been studied before, only approximate solutions were proposed, using heuristic methods [8], [10]. No efficient algorithm that finds the exact solution was proven to exist.

Finding equilibrium strategies for negotiation with multiple divisible issues under time constraints has been proven to be equivalent to solving a particular maximization problem, the *fractional knapsack problem* [11]. The solution can be computed in linear time, in the first negotiation round and the strategy set is a *Nash equilibrium*. We adapt these results to the preference model discussed in this paper. We use a notation similar to [11]. Let $S_i(t)$ be the equilibrium strategy of agent $i \in \{A, B\}$ for $1 \leq t \leq n$.

The following proposition defines *Nash* equilibrium strategy for our bargaining game with deadlines and discounted utility functions.

Proposition 2: For $t = n$, the equilibrium strategy is:

$$S_A(n) = \begin{cases} OFFER\ O_A(n) = \{x | x = \arg\max_y U_A(y, n)\} & \text{if } A\text{'s turn} \\ ACCEPT\ ACC_A(n) & \text{if } B\text{'s turn} \end{cases}$$

$$S_B(n) = \begin{cases} OFFER\ O_B(n) = \{x | x = \arg\max_y U_B(y, n)\} & \text{if } B\text{'s turn} \\ ACCEPT\ ACC_B(n) & \text{if } A\text{'s turn} \end{cases}$$

For $t < n$, the equilibrium strategy is:

$$S_A(t) = \begin{cases} \text{OFFER } O_A(t) & \text{if } A\text{'s turn} \\ \text{IF } \exists x \in O_B(t) \text{ s.t. } U_A(x, t) \geq U_A(t+1) \\ \text{THEN ACCEPT } ACC_A(t) & \text{if } B\text{'s turn} \\ \text{ELSE REJECT} \end{cases}$$

$$SB(t) = \begin{cases} \text{OFFER } O_B(t) & \text{if } B\text{'s turn} \\ \text{IF } \exists x \in OA(t) \text{ s.t. } U_B(x, t) \geq U_B(t+1) \\ \text{THEN ACCEPT } ACC_B(t) & \text{if } A\text{'s turn} \\ \text{ELSE REJECT} \end{cases}$$

where $O_i(t)$ with $i \in \{A, B\}$ are the equilibrium offers of the agents at time t.

$O_A(t)$ is the set of offered constraints satisfied by all solutions x of the following maximization problem:

$$\begin{aligned} &\text{maximize } U_A(x, t) \\ &\text{such that } U_B(x, t) \geq U_B(t+1) \end{aligned} \tag{1}$$

$O_B(t)$ is the set of offered constraints satisfied by all solutions x of the following maximization problem:

$$\begin{aligned} &\text{maximize } U_B(x, t) \\ &\text{such that } U_A(x, t) \geq U_A(t+1) \end{aligned} \tag{2}$$

$U_i(t)$ with $i \in \{A, B\}$ is the maximum utility that agent i can get from his equilibrium offer at time t.

$$ACC_A(t) = \{x \in O_B(t), U_A(x, t) \geq U_A(t+1) | x = \arg\max_y U_A(y, t)\}$$

is the part of the offer of agent B accepted by agent A at time t, if he decides to accept.

$$ACC_B(t) = \{x \in O_A(t), U_B(x, t) \geq U_B(t+1) | x = \arg\max_y U_B(y, t)\}$$

is the part of the offer of agent A accepted by agent B at time t, if he decides to accept.

Proof: The equilibrium strategy is similar to [11], but slightly adapted for the preference model with constraints. At the last step (n), the agent making the offer (say A) is in a strong position and offers a consistent constraint set that gives him the highest utility (diminished with time), i.e. $max(U_A) = max(u_A) \times \delta^{n-1}$. All the values in the domain of the offered consistent constraint set score the same utility. The other agent's (B) best response is to accept, but as specified in Section III, he can accept only a part of the proposal to obtain as much utility as possible. He will intersect the domain of the offered consistent set with one of the consistent constraint subsets of his own constraint set and he will select the intersection that gives him maximum utility, $ACCB(n)$. This intersection of the domains might shrink the domain of the offered constraint set. As agents have different preferences, it might be the case that B gets a high utility, even maximum, making this a key difference from the *split-the-pie* games. At step $t = n-1$, the agent that must make a proposal (B) reasons backwards to $t = n$ and thinks that A will be able to get $max(u_A) \times \delta^{n-1}$. Therefore, at step $t = n-1$, B's best action is to make an offer that will give to A an utility at least as high as $max(u_A) \times \delta^{n-1}$, such that A will immediately accept. From all the possible offers that satisfy this condition, B chooses those that maximize his utility by solving maximization problem (2). The agent that makes the first proposal has complete information about the negotiation setting and therefore at step $t = 1$ he can compute the best offer such that the other agent will accept (the whole offer or only a part of it) immediately by reasoning backwards to $t = n$ and solving maximization problems (1) and (2) for each time step. Both agents play their best response strategies and the strategy set is a Nash equilibrium.

■

The equilibrium solution depends on the agent that makes the first move, so it is neither symmetric, nor unique. As there are two players, A and B, there are two equilibrium solutions. Moreover, the equilibrium solution is a set of multiple possible contracts and the final agreement is established randomly by one of the agents, so there may be many possible agreement contracts. However, as an agent equally prefers any contract from the set of the equilibrium solution, this step is trivial and not of critical importance.

Note that our equilibrium strategy is Pareto optimal, i.e. there is no loss of utility points. At each step, an agent offers to the other agent as much as needed such that he can accept his offer. At the same time he tries to maximize his own utility. The agent that receives an offer is able to accept a part of the offer that gives him the highest utility.

Solving maximization problems (1) and (2) for each time step is a difficult process. Agents must perform suitable combinations of constraints to aggregate a certain utility value. So far in the literature the solutions were to sample the contract space by using heuristic techniques like simulated-annealing [8] thus obtaining near-optimal solutions. In this paper we describe a method to find an exact solution to this problem. The next theorem helps us to reduce the number of combinations in order to find the equilibrium solution.

Theorem 1: If G_{C_i} (corresponding to constraint set C_i) are constraint graphs of agents $i \in \{A, B\}$ then the equilibrium solution is one of the maximal cliques of the constraint graph corresponding to the union of the constraint sets of agents A and B, i.e. $G_{C_A \cup C_B}$.

Proof: The agent that makes the first offer, say A, tries to satisfy as many of his constraints as possible in order to maximize his utility, i.e. a maximal set of constraints. If the offer is constructed according to the method described in Section III, i.e. using the constraints themselves or subintervals of the issue intervals of the constraints, then no additional constraints from his set of constraints, C_A, can be satisfied by the values included in the offer. In conclusion, no other nodes from G_{C_A} are taken into consideration and the equilibrium solution is maximal with respect to the nodes of $G_{C_A \cup C_B}$ that are part only of C_A. The agent receiving an offer, B, can accept a part of the offer that maximizes his own utility, i.e. the one that satisfies as many constraints as possible from C_B. The part of the offer is computed by intersecting the issue intervals in the offer with the issue intervals from his own constraint set. Because no additional constraints can be satisfied by the part of the offer he accepts, it follows that the solution is maximal with respect to C_B (and with respect to the nodes in G_{C_B}). Because no other nodes can be taken into consideration either from C_A or from C_B, it results that the equilibrium solution contains the domain of a maximal consistent set of constraints from C_A and C_B – a maximal clique in $G_{C_A \cup C_B}$. ■

Even if this theorem shows that the search space can be drastically reduced, finding all the maximal cliques of the constraint graph is still a difficult problem. Note however that according to our assumption, constraints are single intervals and the constraint graph is a special type of graph formed by intersecting the issue graphs. Consequently, the following proposition shows that the number of maximal cliques of the constraint graph can be further reduced.

Proposition 3: A clique is maximal in the constraint graph iff its vertex set is the intersection of the vertex sets of maximal cliques in each of the issue graphs.

Proof: Let $G_C = (V, E)$ be the constraint graph and let $G_i = (V, E_i)$, $1 \leq i \leq m$ be the issue graphs. From the definition $E = \cap_{i=1}^{m} E_i$.

$\Longleftarrow$ Let S_i be maximal cliques in G_i and let V_i be their vertex sets, $1 \leq i \leq m$. We define $V_S = \cap_{i=1}^{m} V_i$. Let $x \neq y$, $x, y \in V_S$. It follows that for all $1 \leq i \leq m$ we have $x, y \in V_i$. From the fact that S_i is a clique of G_i it follows that $(x, y) \in E_i$, so $(x, y) \in E$ that clearly shows that V_S induces a clique S of G_C. Let us now prove that clique S is maximal. Assuming by refutation that clique S is not a maximal clique of G_C we would find a vertex $x \in V \setminus V_S$ s.t. $S \cup \{x\}$ is also a clique of of G_C. It follows that $(y, x) \in E$ for all $y \in V_S$, i.e. $(y, x) \in E_i$ for all $1 \leq i \leq m$. Therefore the set of vertices $V_i \cup \{x\}$ would define a clique of G_i, that contradicts the hypothesis that S_i is a maximal clique of G_i.

$\Longrightarrow$ Let S be a maximal clique of G_C and let V_S be its vertex set. From the definition of E as $\cap_{i=1}^{m} E_i$ it follows that S is also a clique of each G_i. We expand S to a maximal clique S_i of G_i and let V_i be its vertex set for all $1 \leq i \leq m$. Clearly $V_S \subseteq \cap_{i=1}^{m} V_i$. If we assume by refutation that the inclusion is strict, we would be able to find another clique S' of G_C with vertex set $V_{S'} = \cap_{i=1}^{m} V_i$ s.t. $V_S \subset V_{S'}$. But this contradicts that S is a maximal clique of G_C. ■

In other words, the maximal cliques of graphs G_{C_A} (G_{C_B}) are obtained by intersecting the vertex sets of the maximal cliques of issue graphs that compose G_{C_A} (G_{C_B}). Note that issue graphs are interval graphs, and interval graphs are also *chordal graphs*[3]. The number of maximal cliques of a chordal graph is at most equal to the number of its vertices [15] ($|C_A|$ for agent A and $|C_B|$ for agent B). It follows that G_{C_A} has at most $|C_A|^m$ (i.e. the total number of intersections) maximal cliques, while G_{C_B} has at most $|C_B|^m$ maximal cliques. This follows from the fact that there are m issue graphs that compose each of the constraint graphs G_{C_A} and G_{C_B}.

Chordal graphs can be recognized in linear time using procedures *Lexicographic Breadth First Search (LexBFS)* [19] and *Maximum Cardinality Search (MCS)* [20]. Both procedures, when applied to chordal graphs, generate a particular ordering of vertices called *Perfect Elimination Ordering (PEO)* (which every chordal graph has [15]). This ordering has the property that any vertex x together with its neighbors that are placed to its right in the ordering ($RN(x)$) form a clique. That is, $x \cup RN(x)$ is a clique. With the help of a PEO it is possible to collect the maximal cliques in linear time [21].

Algorithm 1 computes the equilibrium solution for the case when A is the first mover and runs in polynomial time. The next theorem gives the complexity.

Theorem 2: If each constraint defines a single interval on the real line per issue, the time complexity of finding the equilibrium solution in the first round is $O((n+1) \cdot (|C_A| + |C_B|)^m + m \cdot (|C_A| + |C_B|)^2 + 2 \cdot m \cdot (|C_A| + |C_B| + E))$, where m is the number of issues, E is the maximum number of edges in the issue graphs that compose $G_{C_A \cup C_B}$ and n is the negotiation deadline (the number of steps).

Proof: The result follows by summing up the complexities of individual sections of Algorithm 1.

[3]A graph is chordal if any of its cycles with more than 3 vertices has a chord, i.e. an edge connecting two vertices non-adjacent in the cycle [18].

Algorithm 1 COMPUTE SOLUTION – *A* MOVES FIRST

COMPUTE_A_FIRST(U_A, C_A, U_B, C_B, n)

1: $G_{C_A \cup C_B} \leftarrow$ transformation of C_A and C_B into constraint graph
2: **for** $i = 1$ **to** m
3: $\quad$ $peo[i] \leftarrow ComputePEO(i, G_{C_A \cup C_B})$ for the issue graph i
5: $\quad$ $mc[i] \leftarrow ComputeMaximalCliques(peo[i], G_{C_A \cup C_B})$
6: $solution \leftarrow$ empty clique
7: $MAXCG \leftarrow$ compute maximal cliques of $G_{C_A \cup C_B}$
8: $UAMAX \leftarrow 0$, $UBMAX \leftarrow 0$
9: $LASTUA \leftarrow 0$, $LASTUB \leftarrow 0$
10: **for** $t = n$ **to** 1
11: $\quad$ **if** (t is odd) **then** //A's turn
12: $\quad\quad$ **for each** *clique* **in** *MAXCG*
$\quad\quad\quad$ **if** ($U_A(clique, t) \geq UAMAX$ **and**
13: $\quad\quad\quad\quad$ $U_B(clique, t) \geq LASTUB$) **then**
14: $\quad\quad\quad$ $UAMAX \leftarrow U_A(clique, t)$
15: $\quad\quad\quad$ $LASTUB \leftarrow U_B(clique, t)$
16: $\quad\quad\quad$ $solution \leftarrow clique$
17: $\quad$ **else** // B's turn
18: $\quad\quad$ **for each** *clique* **in** *MAXCG*
$\quad\quad\quad$ **if** ($U_B(clique, t) \geq UBMAX$ **and**
19: $\quad\quad\quad\quad$ $U_A(clique, t) \geq LASTUA$) **then**
20: $\quad\quad\quad$ $UBMAX \leftarrow U_B(clique, t)$
21: $\quad\quad\quad$ $LASTUA \leftarrow U_A(clique, t)$
23: $\quad\quad\quad$ $solution \leftarrow clique$
24: **return** solution

Transformation of constraint sets into a single constraint graph takes $O(m \cdot (|C_A|+|C_B|)^2)$ – line 1. First we create a larger constraint set from the constraint sets of the two agents. The larger constraint set will have size $|C_A| + |C_B|$. Then we create the graph $G_{C_A \cup C_B}$ from this constraint set. The existence of an edge will be tested for every pair of constraints in the set and this operation costs $O(m \cdot (|C_A| + |C_B|)^2)$. The multiplication with m in the equation means that the domains of all issues will be tested for intersection, for each pair of constraints.

LexBFS or MCS algorithms (line 3) take $O(|C_A| + |C_B| + E)$ to produce a PEO that is stored into $peo[i]$ [19], [20]. Maximal cliques are saved into $mc[i]$ in time $O(|C_A|+|C_B|+E)$ [21]. This happens m times, once for each issue graph, so lines 2 to 5 take $O(2 \cdot m \cdot (|C_A| + |C_B| + E))$. Storing maximal cliques of $G_{C_A \cup C_B}$ into $MAXCG$ costs $O((|C_A|+|C_B|)^m)$ by taking the intersections among all possible combinations of maximal cliques of issue graphs. The search for equilibrium solution for each step is carried out in time $O(n \cdot (|C_A|+|C_B|)^m)$. For each step ($n$ in total) we must check all the maximal cliques (at most $(|C_A|+|C_B|)^m$). Summing up, we get $O((n+1) \cdot (|C_A|+|C_B|)^m + m \cdot (|C_A|+|C_B|)^2 + 2 \cdot m \cdot (|C_A| + |C_B| + E))$. ■

We can observe that if the number m of issues is fixed then the complexity is polynomial with respect to the number n of constraints.

V. Related work

According to our literature review, there is little work on negotiation about interdependent issues. Probably the first result to appear in the literature is the negotiation model of Klein [10]. After the authors observed that hill climbers perform better when paired with annealers, they proposed a model with an annealer as a mediator and a voting mechanism for agents. The model performs quite well, but the solutions are not exact and a mediator that performs the annealing is required. Differently from our work, there is no game-theoretic analysis of the model and there is no theoretical study of the preference model.

There are works that build on [10], such as [7]–[9]. All of them use simulated annealing, again without any game-theoretic analysis of the model. Moreover, their model has some disadvantages: the number of bids per agent is restricted because of performance limitations; the achieved optimality decreases with the number of issues. An important thing that worths mentioning is that, while these works used the same preference model as we did (in fact we have borrowed their preference model), they did not derive the same conclusions.

Another work that employs simulated annealing is [6]. Simulated annealing is used by agents to accept contracts and the accepted contracts are mutated in the next step. The contract is thus improved until the deadline is reached. In our work we use the hill climbing strategy for accepting contracts and we compute the exact solution of the negotiation problem.

There are results in the literature that approach the problem differently. By using utility graphs, authors of [5] achieve an exponentially decrease of the complexity of the problem, though the problem remains complex. In [12], the authors consider approximations of generic separable nonlinear utility functions and show that the equilibrium can be computed in linear time. A comparison of the outcomes for different negotiation procedures is also provided.

VI. Conclusions and future work

We have shown that, under certain assumptions, the equilibrium solution of negotiations with nonlinear utility functions can be computed in polynomial time.

We are not aware of a similar work that derives the same powerful conclusions, although a similar preference model (constraints with intervals assigned to issues) has been studied before [7]–[9]. Compared to previous related works, we provide a game-theoretic analysis of the negotiation model using a method inspired from [11] and we show that the equilibrium solution can be computed in polynomial time. The agents are using the hill-climbing approach of accepting contracts. Therefore, we consider that our approach for negotiation with interdependent issues is novel.

As future work, we plan to experimentally evaluate the algorithm in order to see how it performs against real-world complex negotiation scenarios. In particular, we are interested in the average computation time of the algorithm when applied to various realistic scenarios.

References

[1] E. Shakshuki, H.-H. Koo, and D. Benoit, "Negotiation strategies for agent-based meeting scheduling system," in *Proceedings of the International Conference on Information Technology*, ser. ITNG '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 637–642.

[2] J. Wainer, P. R. Ferreira Jr., and E. R. Constantino, "Scheduling meetings through multi-agent negotiations," *Decis. Support Syst.*, vol. 44, no. 1, pp. 285–297, November 2007.

[3] Y. Jin and M. Geslin, "Roles of negotiation protocol and strategy in collaborative design," in *Design Computing and Cognition '08*, J. S. Gero and A. K. Goel, Eds. Springer Netherlands, 2008, pp. 491–510.

[4] K. P. Clark, S. van Splunter, M. Warnier, and F. M. T. Brazier, *Expressing intervals in automated service negotiation*, ser. Grids and Service-Oriented Architectures for SLAs. Springer-Verlag, 2010, ch. 7, pp. 67–75.

[5] 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*. New York, NY, USA: ACM, 2005, pp. 280–287.

[6] A. Kardan and H. Janzadeh, "A multi-issue negotiation mechanism with interdependent negotiation issues," in *ICDS '08: Proceedings of the Second International Conference on Digital Society*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 55–59.

[7] H. Hattori, M. Klein, and T. Ito, "A multi-phase protocol for negotiation with interdependent issues," in *IAT '07: Proceedings of the 2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 153–159.

[8] T. Ito, H. Hattori, and M. Klein, "Multi-issue negotiation protocol for agents: Exploring nonlinear utility spaces," in *IJCAI*, M. M. Veloso, Ed., 2007, pp. 1347–1352.

[9] K. Fujita and T. Ito, "A preliminary analysis of computational complexity of the threshold adjusting mechanism in multi-issue negotiations," in *WI-IATW '07: Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 383–386.

[10] M. Klein, P. Faratin, H. Sayama, and Y. Bar-Yam, "Negotiating complex contracts," *Group Decision and Negotiation*, vol. 12, pp. 111–125, March 2003. [Online]. Available: http://jmvidal.cse.sc.edu/library/klein03a.pdf

[11] S. Fatima, M. Wooldridge, and N. R. Jennings, "Multi-issue negotiation with deadlines." *Journal of Artificial Intelligence Research*, vol. 27, pp. 381–417, 2006. [Online]. Available: http://eprints.ecs.soton.ac.uk/13079/

[12] S. S. Fatima, M. Wooldridge, and N. Jennings, "An analysis of feasible solutions for multi-issue negotiation involving non-linear utility functions," in *Proc. 8th Int. Conf on Autonomous Agents and Multi-Agent Systems*, 2009, pp. 1041–1048. [Online]. Available: http://eprints.ecs.soton.ac.uk/17065/

[13] S. Fatima, M. Wooldridge, and N. R. Jennings, "Approximate and online multi-issue negotiation." in *6th International Joint Conference on Autonomous Agents and Multi-agent Systems.*, 2007, pp. 947–954. [Online]. Available: http://eprints.ecs.soton.ac.uk/14219/

[14] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton, NJ: Princeton University Press, 1947.

[15] D. Fulkerson and O. Gross, "Incident matrices and interval graphs," *Pacific J. Math*, pp. 835–855, 1965.

[16] M. Osborne and A. Rubinstein, *A Course in Game Theory*. Cambridge, MA: MIT Press, 1994.

[17] A. Rubinstein, "Perfect equilibrium in a bargaining model," *Econometrica*, vol. 50, no. 1, pp. 97–109, January 1982. [Online]. Available: http://ideas.repec.org/a/ecm/emetrp/v50y1982i1p97-109.html

[18] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57)*. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., 2004.

[19] D. J. Rose, R. E. Tarjan, and G. S. Lueker, "Algorithmic aspects of vertex elimination on graphs," *SIAM J. Comput.*, pp. 266–283, 1976.

[20] R. Tarjan and M. Yannakakis, "Simple linear-time algorithm to test chordality of graphs, test acyclicity of hypergraphs, and selectivity reduce acyclic hypergraphs," *SIAM J. Comput. 13*, p. 566579, 1984.

[21] J. R. Blair and B. Peyton, "An introduction to chordal graphs and clique trees," *IMA*, vol. 56, pp. 1–29, 1993.