An agent architecture for simultaneous bilateral negotiations

Angela Fabregues and Carles Sierra
IIIA: Institut d'Investigació en Intel·ligència Artificial
CSIC: Spanish Scientific Research Council, UAB
08193 Bellaterra, Catalonia, Spain
{fabregues, sierra}@iiia.csic.es

ABSTRACT

In this paper we introduce an agent architecture that is suitable for joint action plan negotiation among several agents in complex environments and with negotiation time bounds. The architecture is based on a graded BDI model to drive the decisions that the agent makes [4]. The practical reasoning explores the space of possible deals with the help of a genetic algorithm that copes with the potentially sheer amount of joint plans. Beliefs represent the agent's view of the world. Before starting negotiating the agent summarises all what she knows into the five dimensions of the LOGIC negotiation model [12]. This summary is used to decide what to say next and whom to say it to in order to sign deals on joint plans. This work does not yet include experimental results but it is going to be tested using the DipGame testbed (`http://www.dipgame.org`).

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, Human Factors

Keywords

agent architectures, simultaneous bilateral negotiation, dipgame, diplomacy game

1. INTRODUCTION

As stated in [10], a lot of research work on automated negotiations have appeared in recent years but very few of them consider negotiations with humans. Most of the papers on automated negotiation focus on bilateral negotiations where software agents compete using utility maximisation strategies. These strategies may work well when negotiating with software agents but not necessarily against humans that in many cases will not behave according to a constructivist rationality principle [1]. For instance, human agents' decisions depend a lot on e.g. the relationships with the other agents and on emotions. In this work we deal with competitive environments with simultaneous bilateral negotiation with repeated encounters. The other agents can be human or software. And they are able to co-operate performing joint plans to get a better outcome. Here, as we will see, good does not exclusively means high utility but also other desired features.

We assume that agents are negotiating over large spaces of potential joint plans, in a limited amount of time and with limited resources. These are the usual settings that humans are faced with in their everyday life. Humans do not attempt to find the optimal solution before starting negotiating with others. They work well with uncertain environments and do not doubt to negotiate a plan that is not optimal if they think it is good enough. Usually, it is not possible to get the optimal one because of time constraints and because of the competitive simultaneous negotiation setting.

The architecture proposed in this paper is specially designed for environments where several self-interested agents compete performing actions periodically in specific time instants. All the agents perform the actions at the same time. Actions change the state of the world but not all the actions are successfully executed, i.e. there is uncertainty on whether an action will be successful. Actions of all the agents are executed concurrently, therefore conflicts arise between actions and some of them will fail.

Before selecting what actions to perform, agents interact with the aim of deciding joint plans, that is, to co-operate. Agents are assumed to be competitive but they are also interested in increasing their confidence on the other agents' next actions. This is because knowing what the other agents are going to do allows us to choose actions that avoid undesired conflicts. Getting from others this information and deciding joint plans is possible thanks to negotiation. In the environments we envisage an agent negotiates trying to convince the others to perform the set of actions that it desires them to perform. There would be few actions to execute but a lot of negotiation to decide them.

Agents can observe the state of the world and the other agents actions independently of whether those were successful or not. On the other hand, negotiations are private. An agent cannot even know who is negotiating with whom when it is not involved. Also the personality and the beliefs, desires and intentions of the other agents are unknown and can only be guessed, somehow, from the agent's behaviour. For example, if someone assures us that she will perform an action different to the one she finally performs, we can assume that this agent was not trustful. She lied us or at least she changed her mind without telling us. As agents negotiate repeatedly forming close groups of agents, they can build models on the other agents' personality and behaviour based on previous experiences. These models will help a lot to avoid agreeing on joint plans with untrustworthy agents.

This paper is ongoing work and there are not experimental results yet but it is planned to test the architecture using the

DipGame testbed even against human players. DipGame provides an environment complex enough for testing this work. And through its site (`http://www.dipgame.org`), human players are able to negotiate with our agents and thus test our work against humans.

In this work we first shortly introduce, as background material, the g-BDI and LOGIC (Section 2) models that we incorporate as components of our architecture. We describe the architecture (Section 3) to later explain it in detail in several sections: the communication (Section 4), the practical reasoning (Section 5), the negotiation (Section 6) and the strategy (Section 7). Then we discuss the importance of trust in this architecture (Section 8) and introduce the DipGame testbed (Section 9). The paper ends with a discussion about future work (Section 10).

2. G-BDI AND LOGIC

Taking a look into the literature we see that one of the most popular agent models is the BDI model [9, 11, 2]. BDI architectures have Believes, Desires and Intentions as their principal components. The BDI model has been used to cope with complex, dynamic and uncertain environments. This agent model is based on the theory of human practical reasoning that deals with the deliberation on what actions to perform.

We use a graded BDI model, g-BDI [4], in order to describe the mental state of the agent with degrees. We want to build agents capable to interact with humans. And human agents are really good working in uncertain environments. We point out that this graded BDI model will be very useful for dealing with complex environments where the world is not perfectly observable as the ones we are facing with here (e.g. the performance of an action by an agent is uncertain).

The other model that we base our work on is the LOGIC negotiation model [12]. This work points out the information that we need in order to be able to negotiate with humans in repeated encounter scenarios. That information is organised into five dimensions:

- **Legitimacy**. The relevant information that might be useful to the other agent in the negotiation.
- **Options**. Deals that are acceptable.
- **Goals**. To achieve our desires.
- **Independence**. Outside options. What can we do if the negotiation fails.
- **Commitments**. Previously signed deals, C.

We use this model but not exactly as it is described in [12]. The LOGIC dimensions are assumed to be prepared before every single negotiation with an agent. Then the agents negotiate exchanging several proposals until a deal has been accepted or someone withdraws. We do this differently. We do not select the negotiation partner beforehand. We maintain the information on the five dimensions updated all the time and negotiate with one or another depending only on the deal we want to sign. In this way, the partnership selection is implicit in the selection of the next option to negotiate. This is because our architecture allows for multi-lateral negotiations while LOGIC was designed only for bilateral ones. We are also interested in the way that LOGIC proposes to evaluate the relationships between the different agents. Specially the classification of relationships as in need, in equality and equity are very useful in our architecture because it models very well actual human relationships.

3. ARCHITECTURE

The negotiation architecture that we propose in this work is based on a graded BDI. As we can observe in Figure 1, the agents desires are the central part of all the architecture because they allow to evaluate the agents satisfaction in a certain state, either the current state or a simulated one. The agent desires change along time and it is the strategy of the negotiation that determines how. Each agent has its own strategy that depends on its personality traits. For example, if we have a very conservative agent, then her strategy will change her desires smoothly.

All the negotiation is about is satisfying the desires of the agent. The intentions are fed by the desires. The desires provide decision functions concerning what to do, what to say and when to do so.

The beliefs of an agent are a summary of the information on past observations and interactions with other agents that she has stored in the history. Every time a message or a change in the environment is observed, it is stored in the history. But to be able to update our beliefs we use an interpreter that is a mechanism capable of inferring facts based on the information that is stored in the history. These facts with their corresponding degree are the beliefs of the agent. The interpretation of the environment observations (i.e. the perception) depends mostly on the agent itself. We assume that an agent has a set of local rules to interpret the dialogical actions. The rules determine updates in the beliefs model of the agent. The schema of those rules is:

$$\text{State}(\sigma) \text{ AND } \text{Message}(\mu) \text{ THEN } \text{Update}(\mathcal{B}(\varphi))$$

and the intended meaning of a rule is: from the current environment and internal states and a perceived message then update the beliefs model. Every negotiating agent will then possess a theory consisting of a number of such rules that will update the agent's model on the behaviour of other agents.

Our actions depend on our intentions, but also on what is possible to do. The agent tries to identify a set of actions that will lead to a good enough solution. From a set of plans the negotiation options are formulated as deals to be signed with other agents (or humans) in order to allow us to be sure that our plans will succeed.

This architecture is been build to be used in complex and very competitive environments as those introduced before. In those environments, agents are not able to wait until the optimal solution is found. They have to interact with other agents from the beginning, and make proposals of deals on joint plans while they decide which plans are the best ones. Thus, the execution of an agent consists on several concurrent processes: one for message listening (to receive proposals, rejections, or acceptances), another to observe the changes in the environment (similar to the message listener), another for practical reasoning (to generate joint plans), one for negotiating (selecting proposals to make) and one more for each process that updates the several boxes that appear in the architecture.

The data in the boxes is constantly being updated as these processes are running all the time. But the negotiation pro-

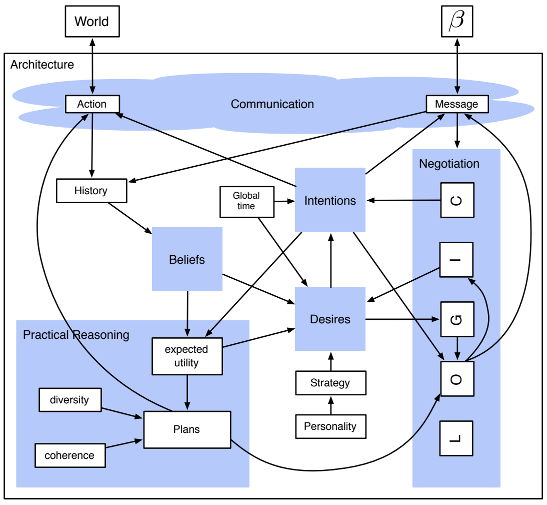

Figure 1: Agent architecture. Boxes represent data and arrows represent processes.

cess gets idle regularly when a proposal is sent to another agent as we want to have only one proposal open at anytime. This simplifying restriction will be removed in future versions of the agent implementation. The idle time gets over when an answer from the appropriate agent is received or too much time has passed without any answer. See Algorithm 1 for a multithreaded version of the agent. The meaning of the different variables will become clear when reading the subsequent sections.

The algorithm consists on initially spawning all the processes including practical reasoning. And then, repeat a sequence of: a number of negotiation rounds (modelled as a function) up to the time limit, a selection of actions from the set of agreed upon joint plans and their execution.

Algorithm 1 function init(t_{max})

```
Require: shared $P^t$ {current plans}
Require: shared $\delta = \bot$ {current proposed deal}
Require: shared $response = \bot$ {received response to $\delta$}
Require: shared $patience = default$ {patience}
  spawn $pr_1 ... pr_n$ {processes associated to the architecture diagram arrows}
  spawn $practicalReasoner()$
  while $\top$ do
    $t_{limit} \leftarrow t_{now} + t_{max}$
    while $t_{now} < t_{limit}$ do
      $negotiation()$
    end while
    $Actions \leftarrow selectActions(P^t)$
    $perform(Actions)$
  end while
  return
```

The negotiation function starts waiting. The practical reasoning thread needs some time to get enough plans to feed with options the negotiation process and time must be also given to other agents to analyse our proposals after sending them. When new messages arrive, we check if it is a proposal. If it is, the message is stored in the set of proposals. This set is shared with the negotiating function that periodically checks this set to select a subset of proposals that can be jointly acceptable. The rest of proposals are rejected. If none of the proposals is good enough, a new deal is selected and the negotiation round is finished.

$$
\begin{array}{l}
\mathcal{L}_1 ::= \text{propose}(\alpha, \beta, deal) \mid \text{accept}(\alpha, \beta, deal) \mid \\
\text{reject}(\alpha, \beta, deal) \mid \text{withdraw}(\alpha, \beta) \\
deal ::= \text{Commit}(\alpha, \beta, \varphi)^+ \mid \text{Agree}(\beta, \varphi) \\
\varphi ::= \underline{predicate} \mid \text{Do}(\alpha, \underline{action}) \mid \varphi \wedge \varphi \mid \neg\varphi \\
\beta ::= \alpha^+ \\
\alpha ::= \underline{agent}
\end{array}
$$

Figure 2: $\mathcal{L}_1$ language definition.

Every proposal is stored in δ until an answer is received or a timeout fires. The negotiation function, Algorithm 2, waits for this answer. When it arrives, it stores it in the variable *response* and the negotiation function resumes its execution by processing the answer.

Algorithm 2 function negotiation()

```
$waituntil(t_{now} = patience$ or $response \neq \bot)$
if $\delta \neq \bot$ then
  if $response = \bot$ then
    $sendWithdraw(\delta)$
    $\delta \leftarrow \bot$
  else
    if $response = accept(\delta)$ then
      $Commitments \leftarrow Commitments \cup \{\delta\}$
      $response, \delta \leftarrow \bot$
    end if
  end if
end if
$Proposals' \leftarrow Proposals$
$Proposals \leftarrow Proposals \setminus Proposals'$
$Deals \leftarrow getProposalsToAccept(Proposals')$
for $\delta' \in Deals$ do
  $sendAccept(\delta')$
end for
for $\delta' \in Proposals' \setminus Deals$ do
  $sendReject(\delta')$
end for
if $Deals = \emptyset$ then
  $\delta \leftarrow selectNextDeal(Plans)$
  if $\delta \neq \bot$ then
    $sendProposal(\delta)$
    $patience \leftarrow t_{now} + t_{max}^{\beta(\delta)}$
  end if
end if
return
```

In the subsequent sections we explain every component of the architecture and provide the equations for the functions that appear in the already introduced algorithms.

4. COMMUNICATION

Agents interact with the environment (performing actions) and also with other agents (sending and receiving messages). Actions and messages need to be understood by all the participants of the communication. The communication module provides a language useful for formulating actions and also proposing deals. We assume that a common language is shared. We use the $\mathcal{L}_1$ language that is the first level of the language tower proposed in [8]. It allows agents to interact exchanging proposals as defined in Figure 2. For a correct understanding of the dialogues that we can generate with this language it is necessary also to share a protocol like the one represented in Figure 3. The protocol regulates the interaction between agents letting the agents make proposals, accept or reject them.

This protocol enforces that every proposal message will be answered by either an accept or a reject. When no answer is

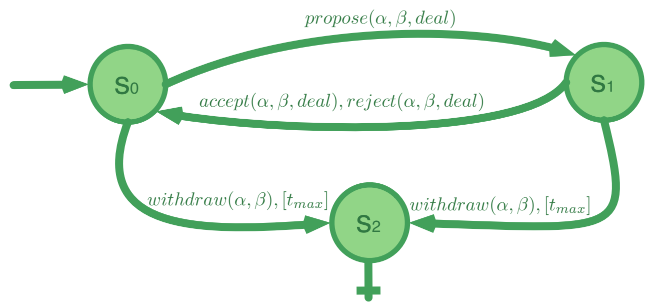
$propose(\alpha, \beta, deal)$
S0
S1
$accept(\alpha, \beta, deal), reject(\alpha, \beta, deal)$
$withdraw(\alpha, \beta), [t_{max}]$
S2
$withdraw(\alpha, \beta), [t_{max}]$

Figure 3: Communication Protocol for $\mathcal{L}_1$.

obtained in a certain interval of time, t_{max}, the negotiation is over and a withdraw message is sent.

5. PRACTICAL REASONING

Practical reasoning concerns the deliberation that precedes a decision making on what actions to perform. This deliberation in our architecture provides a set of action plans that can include actions to be performed by other agents. Those plans that involve actions for more than one agent are joint plans and require the collaboration among agents.

Plans can be composed by other plans forming a hierarchy. A basic plan is a tuple (α, a, t) formed by an agent α, an action a and an instant of time t meaning that the agent α would perform action a at time t. A plan is ultimately a set of such basic plans: the leaves of the plan tree. The set of possible hierarchical plans is denoted by P. The set of basic plans is denoted as B and the space of possible plans is then $P = 2^B$. The empty set meaning inaction. We denote the set of basic plans of a plan p as $basic(p) \subseteq B$.

A plan is *part of* another plan if and only if every basic plan in the first is a basic plan of the second, that is: p part_of $p' \iff basic(p) \subseteq basic(p')$. A plan is *unfeasible* if it has two or more incompatible actions. The incompatibility of actions depends on the application domain, e.g. in the Diplomacy game we cannot move the same unit to two different regions at the same time. Two feasible plans $p, p' \in P$ are incompatible if $basic(p) \cup basic(p')$ is an unfeasible plan.

We evaluate single plans based on the utility that they provide. We assume that there is a utility function $U : P \times W \to [0, 1]$ for each plan and world state, and a confidence measuring the probability that the whole plan would be executed, $C : P \to [0, 1]$. Note that as some actions may depend on other agents we may be uncertain about them being actually performed.

DEFINITION 1. *Given a plan $p \in P$, the* value *of plan p in world state w, $\mathcal{V}(p, w)$, is defined as the expected utility of the plan given the probability (confidence) of the plan being executed:*

$$\mathcal{V}(p, w) = E[\mathcal{U}(p, w)] = \mathcal{U}(p, w) \cdot \mathcal{C}(p) \tag{1}$$

The evaluation of the confidence and the utility depend a lot on the application domain and on the beliefs of the agent. We measure the confidence as the probability that we have on all agents involved in the plan performing the actions that the plan requires them to perform. That is, as our degree of belief on it (see [4] for details on the modelling of beliefs). If an agent thinks that it is not possible for a plan to be performed because the other agents are not going to do what we want to, then it will not take it into consideration and will not engage in negotiations over it, the agent will just look for another option.

The confidence level of a plan is usually in line with the trust on other agents and on the time horizon set by the latest basic plan in the plan. The more agents the plan depends upon, the lower the probability of success and the latter the basic plan to be completed the more improbable the plan is as the probability of the environment changing (and thus making the actions unfeasible) in the meantime will be higher.

The utility of a plan is calculated based on the utility of the states of the world that we can get executing it and the probability on each of those states being reached.

DEFINITION 2. *Given a plan $p \in P$, the* utility *of executing it in the current world state w, $U(p, w)$, is defined as the expected utility of the world states that we can reach performing p:*

$$\mathcal{U}(p, w) = \sum_{w_j \in W} \mathbb{P}(w_j | p, w) \times \mathcal{U}(w_j) \tag{2}$$

Both, the utility and the probability of reaching a world state ($\mathcal{U}(w_j)$ and $\mathbb{P}(w_j | p, w)$) depend on the application domain. In domains where the world is not perfectly observable and thus we are not sure about the current state nor about the state transition probabilities, we could use Partially Observable Markov Decision Processes (POMDPs). However, a wealth of data would be necessary to model it accurately.

Even when the world is perfectly observable, we have to deal with uncertainty because we do not know what the other agents are going to do. And their actions will affect the world as well.

The utility takes into account the uncertainty on the world. Contrarily, the confidence measures to what extend we are sure that the plan will be executed.

DEFINITION 3. *Given a plan $p \in P$, the* confidence *of executing it, $\mathcal{C}(p)$, is defined as the degree of belief on all its basic plans $b \in basic(p)$ being executed:*

$$\mathcal{C}(p) = \prod_{\substack{b \in basic(p) \\ \mathcal{B}(b, r_b)}} r_b \tag{3}$$

Before starting a negotiation process, our belief on a basic plan being executed can be low. An agent can negotiate to increase its confidence on it. This is done by signing deals that provide commitments on other agents actions. When an agent commits to the execution of a basic plan, this commitment is added to the set of current commitments, C and our belief on this basic plan being executed is the trust that we have on it [5]. But to decide whether to negotiate or not a deal on a basic plan, we can estimate the confidence that we would obtain after negotiating it.

DEFINITION 4. *Given a basic plan $b = \langle \alpha, a, t \rangle$, the estimation of belief on its future execution $\mathcal{B}(b, r_b)$ if we engage in a negotiation, is defined as:*

$$r_b = \mathbb{P}(b | \neg\varphi) \times \mathbb{P}(\neg\varphi) + \mathcal{T}(\beta, \alpha, \varphi) \times \mathbb{P}(\varphi) \tag{4}$$

where $\varphi = Commit(\alpha, \beta, Do(\alpha, \langle a, t \rangle))$, and $\mathcal{T}(\beta, \alpha, \varphi)$ is the trust that β has on α honouring the commitment.

The search space for plans is exponential in the number of possible actions and the evaluation function changes so much due to changes in the environment that we have to work with good enough plans instead of waiting for an optimal solution. In the kind of environments we are working on, we assume optimality is never achievable in practical terms. Therefore, the usual planning techniques do not provide good results. What we need is a practical reasoning engine that would provide a set of feasible plans at any time taking into account the dynamics of our beliefs. It is important to have a rich set of solutions instead of just one because it makes the negotiation more robust providing several options for negotiation.

We would like that the set of plans satisfy a trade-off between coherence and diversity. That is, we want the plans to be different —not all depending on the same set of basic plans, that is we want high diversity, but also to be related —to have some basic plans in common with other plans, that is to have some coherence in the set of plans. We thus define the similarity of two plans as the proportion of basic plans that they have in common.

Definition 5. *The similarity between two plans $p, q \in P$, is defined as the proportion of common basic plans:*

$$sim(p,q) = \frac{basic(p) \cap basic(q)}{basic(p) \cup basic(q)} \tag{5}$$

To define a degree of trade-off between coherence and diversity we use the Gini mean difference as a measure of dispersion that compares pairs of elements, in our case plans, of a set.

Definition 6. *Given a set of plans P^t we define the* statistical dispersion *of P^t, $\Delta(P^t)$, as the Gini measure of the plans similarity.*

$$\Delta(P^t) = \frac{1}{|P^t|(|P^t|-1)} \sum_{p,q \in P^t} sim(p,q)$$

Thus the higher the gini value, the higher the dispersion and the lower the gini the higher the coherence. The problem is that gini compares pairs of plans, not sets of them. Then it could happen that a set of plans with high dispersion could have a single basic plan appearing in all the plans of the set. This is not desirable because all the plans would depend on that single basic plan. This is a problem that could be solved applying a measure of dispersion not only to pairs but also to subsets of three, four or more plans. But the complexity of the resulting measure would be too high. Instead of this we use a measure of the popularity of the basic plans that we combine with the Gini dispersion measure. Then, the lower the maximum popularity of basic plans, the better.

Definition 7. *The* popularity *of a basic plan $b \in B$ within a set of plans P^t is the proportion of plans in P^t that contain it:*

$$pop(b, P^t) = \frac{|\{p \in P^t : b \in basic(p)\}|}{|P^t|} \tag{6}$$

The relevance of having a set of plans coherent and diverse will be better understood when introducing the concept of deals in the next section. In addition to looking for diversity of basic plans we could also look for diversity of negotiation partners (the agents involved in the basic plans), of the actions or of the time. Or whatever other dimension of a basic plan if they were defined with more dimensions.

We use genetic algorithms (GA) to look for those plans. Genetic algorithms are a potential technology to explore large spaces. They produce successive populations of solutions that are increasingly better adapted to the environment even when it is dynamically changing along time. For us, each single solution, a chromosome in the GA, represents a feasible plan. The GA will then work with a population of feasible plans. This population is updated generation after generation by the crossover, mutation and selection operators. Crossover and mutation must guarantee the feasibility of the generated plans.

We look for a set of plans being coherent and diverse, as explained before. Therefore, we use an elitist selection over some individuals that will survive to the next generation. Elitism should be applied to the subset of the population that better satisfies a combination of factors: being coherent, diverse, having a minimum number of basic plans, and having substantial utility and confidence. To facilitate the coverage of the whole search space, also other individuals survive but with a probability of survival proportional to the value of their fitness. Finally, the rest of the members of the new population are generated with the crossover mechanism. A certain probability of mutation allows to explore plans containing basic plans that were not included in the initial population.

As explained before, the environment is constantly changing and thus the beliefs of the agent. As the confidence and utility measures depend on it, the individual evaluation of the plans can change drastically and thus provoke an *apparently* incoherent behaviour of the agent. Like proposing to do the opposite that was proposed seconds ago. If we guarantee that a set of *coherent* plans survive to the next generation, the changes in the plans that are available for negotiation is not so drastic and therefore the behaviour of the agent will not be too erratic. On the other hand, a certain degree of diversity allows the search to explore the whole space of plans. This permits the agent to be prepared for any changes in the environment when they happen. Just in the same way as diversity acts in nature. Adjusting the level of coherence and diversity is thus crucial to have a good set of plans available at any time.

The specific design including the codification of plans into chromosomes as well as the way to build the first population, and how to apply crossover and mutation depend, obviously, on the application domain.

5.1 Select actions to perform

The practical reasoning module maintains at any time a set of diverse and coherent plans. We will negotiate using the available plans by proposing deals based on them. But at the end of the negotiation time, the surviving plans (that have not failed in the negotiations) will be used to select the actions to perform as already mentioned in algorithm 1.

When the time for negotiating is over, the agents execute their actions concurrently. We could check the set of remaining plans, choose the plan with highest value and execute those basic actions of the plan that correspond to us. However, other information can also be taken into account: the previous commitments with other agents, or the kind of relationship we would like to build up with the other agents.

For example, if we do not care about the relationship with someone, then we will not care about not honouring previous deals with that agent. This information is part of the intentions. If we have the intention on non respecting a given deal, the commitments that emerge from that deal will not be taken into account when selecting the actions to perform.

In repeated negotiation encounter scenarios, agents form an image on the other agents based on their behaviour. They analyse what the others do and try to discover what the others know in order to be able to predict their actions. To avoid a deterministic behaviour of the agent that would make us vulnerable in front of disloyal attacks, it is better to select the plan with a random method giving more chances to the plans that are better. For example, the plan with the highest value would have more chances to be selected.

6. NEGOTIATION

Agents' plans of action usually contain basic plans with actions assigned to other agents. Those plans are considered joint plans. Plans and joint plans are formulated by the practical reasoning module but the other agents involved in those plans do not know about them. Or, at least, do not know about our interest on them. Thus, the confidence that we have a priori on those plans is not good at all. We would like that the other agents commit to the performance of the actions associated to them in the highest utilitarian plans because the commitments increase the confidence in their eventual execution.

The first thing we need to do is to check which are our options for negotiation. Then, we evaluate them taking into account our goal in the negotiation, the previous commitments that we have and the independence[1]. In the following we describe in more depth those dimensions, their usage in the model and how to formulate the negotiation options.

The options in a negotiation are the deals that we can accept [12]. A deal is in fact a plan. The signature of a deal obliges[2] the agents accepting the deal to perform the actions that the plan assigns to them. The practical reasoning module provides us a set of plans that would be able to be used as deals. But it is sometimes necessary to split those plans into several deals as they need to be negotiated separately with different agents. Thus, from the plans we produce deals that have actions assigned to our agent and just another agent per deal. We denote the set of deals produced by a plan p as a set of plans: $deals(p) \subset 2^p$.

When a deal has been accepted, the negotiating agents commit to perform the actions in the basic plans that are assigned to them. Commitments being honoured up restrict the behaviour of agents. Achieving commitments from other agents is essential as their behaviour get restricted and then the confidence for our plans increases[3]. At the same time, our commitments restrict our behaviour making us more predictable and thus vulnerable from being exploited. After signing a deal, some of the previous possible plans get incompatible with the current commitments (the set of commitments is in fact a plan). Unfeasible plans are not considered when computing deals.

[1]The independence of an agent is measured as the current number of plans available to the agent.

[2]Although we do not assume a normative environment with sanctions for commitment violation.

[3]As the success of the execution of our actions depend on the actions that other agents perform.

A deal is *good* if it provides satisfaction to the agent. We thus evaluate a deal as the estimated satisfaction of the agent desires when adding this deal to the set of commitments, $\delta \cup C^t$.

Agents want to get the best possible set of commitments, that is, the set that is compatible with the plans that have the highest utility. At the same time, the confidence in those plans has to be high as to guarantee their execution. And also the set of available plans, i.e. independence, has to be kept as large as possible to allow for further improvement of the current set of commitments. Therefore, the satisfaction of an agent is defined as an aggregation operator over three criteria: utility, confidence and independence. This aggregation operator may change along time. For instance, we may initially want as much utility as possible over a confidence threshold but later we are more interested on keeping some independence. Agents would desire to have a significant degree of independence to be able to improve the outcome of a concrete negotiation round via negotiations with other agents or just to avoid showing a predictable behaviour (i.e. not showing the others that the agent has no alternative plan to the one being negotiated).

DEFINITION 8. *Given a set of plans P^t and a set of commitments C^t, the* satisfaction *of an agent is defined as an aggregation of three different criteria: utility, confidence and independence.*

$$sat(P^t, C^t) = f^t(v_{\mathcal{U}}, v_{\mathcal{C}}, v_{\mathcal{I}}) \tag{7}$$

where:

- *f^t is an aggregation operator. See for instance [3].*
- *$v_{\mathcal{U}}$ is the OWA average utility of the plans in P^t.*
- *$v_{\mathcal{C}}$ is the average confidence of the plans in P^t.*
- *$v_{\mathcal{I}}$ is the percentage of plans in P^t compatible with C^t.*

The coherence and diversity that we were looking for when searching for plans is crucial to be able to obtain an interesting set of deals. If the coherence of the plans is high, we will have options that do not restrict a lot our independence. When we sign a deal that is incompatible with a plan, we lose the opportunity to achieve it. And thus, the opportunity to use the deals that are only provided by those plans that are now incompatible. In this way, if we are not capable of convincing someone on accepting to perform one of the basic actions of our preferred plan, then we will still be able to use the already signed deals coming from this plan to satisfy other plans that are good as well.

The commitments are just the union of all the deals that are already accepted. Those deals can be jointly incompatible. This would mean that someone will deceit. Trust will be useful to estimate who will deceit and to decide if we should honour or not a deal.

Finally, legitimacy is not explained here because we focus on the exchange of deals and not other types of information. In future work we will use upper language levels in the tower [8] that allow for information exchange and then the legitimacy dimension will become important.

7. NEGOTIATION STRATEGY

The agent desires drive all the negotiation process because the deals are evaluated as the satisfaction that the agent would experiment if she signs the deal. Then the strategy of the agent is modelled in our architecture as rules on how the desires of the agent should change. For example, we usually want as much utility, confidence and independence as possible. But while we aspirate to increase the utility and confidence during the negotiations, we know that the independence will decrease as time goes by. As the negotiation process arrives to the end, the number of signed deals increases and thus the number of commitments. And those commitments that are going to be honoured up restrict the behaviour of the agent reducing her independence.

The desires of the agent change along time in accordance to the strategy. Thus, the satisfaction of an agent changes even when the plans P^t and the commitments C^t remain temporally constant. The personality of the agent determines the strategy to use. For example, a paranoid agent will always give a lot of relevance to the independence because it is incapable to feel confident when other agents know what she is going to do. Contrarily, someone obsessed with taking as much utility as possible will ignore the independence criteria and will focus on maximising the utility. The more cautious the agent is, the more relevance it will put on the confidence thus it will ensure the correct execution of the plans. We point out that the best strategy is to balance this trade-off conveniently. We need independence, but we also need confidence in the most utilitarian plans.

The way to make this process operational is by means of the *satisfaction aspiration* that defines a threshold for a deal being considered acceptable during a time-bounded simultaneous negotiation process with several agents.

DEFINITION 9. *Given a set of feasible plans P^t, and a set of commitments C^t, the* satisfaction aspiration *of an agent at time t, noted $\mathcal{A}(t)$, is defined as*

$$\mathcal{A}(t) = min(t) + \left(\frac{t_{limit} - t}{t_{limit}}\right)^{\tau} \cdot (max(t) - min(t))$$

where $min(t) = sat(P^t, C^t)$ is the current satisfaction of the agent, $max(t) = max_{p \in P} sat(P^t, C^t \cup p)$ is the maximum achievable satisfaction, t_{limit} is the time limit, and $\tau \in [0, 1]$ is the aspiration decay rate.

The negotiation strategy manages the negotiation goals of the agent and decides which deals are acceptable based on the current satisfaction aspiration level. The aspiration level of an agent at a given time depends on the satisfaction obtained from the currently signed deals and on the maximum potentially achievable satisfaction, which is the one obtained from the combination of the signed deals and the best available plan. What we aspire to obtain is always higher than what we have, but we are less optimistic as the negotiation gets closer to the end.

When a proposal arrives, we decide to accept it or not depending on whether it is acceptable —has an estimated satisfaction higher than $\mathcal{A}(t)$, or not. Also, for selecting deals to propose we choose just one randomly between the set of deals that are acceptable, i.e. over $\mathcal{A}(t)$.

8. TRUST

When people talk, negotiate and sign agreements, they sometimes lie with the aim of getting a better immediate outcome. By committing ourselves to do some actions we can convince the other agents to accept deals that would not be accepted otherwise. We have always the option of non respecting the previously signed deals. But we have to be careful because disloyal behaviour cause that the other agents would form a non trustful image of us and they would reject negotiating with us any more.

Trust is crucial for any negotiation to happen. The aim of a negotiation is to obtain the commitment of another agent on performing a given action. But, if we do not trust the other agent and thus we think that she is not going to honour up her commitments, then it has no sense to start negotiating with her. The same happens if we think that the other agent is not capable of doing what we are proposing her to do. In this way, we model trust as an expectation on the execution of a commitment as was proposed in [5] and applied to partnership selection in [6]. It is based on entropic measures applied over the history of interactions.

In repeated negotiation encounter scenarios, the agents interact often and build relationships among them. The trust in those environments is usually high because of the existence of an interest in the continuity of the relationships. This interest makes it worth it to be trustful. If the relationship has not an apparent ending, then it seems that is better to obtain few outcome now but a lot of times of it than to get a large outcome now but just once. We have to take also into account that agents could later on retaliate.

9. DIPGAME

DipGame is a multiagent systems' testbed developed at the IIIA-CSIC. It is based on Diplomacy, a popular strategy board game in which seven players negotiate in order to decide joint plans. The goal of the game is to conquer Europe. To better progress in the game you need to perform actions that are *supported* by other players. Joint plans then usually contain supporting actions between the players. Therefore, although players compete to win they sometimes co-operate. Deciding when to co-operate and persuading the others to support what you want to do, are the most important skills that a player needs to have in order to win the game. All the players' actions are executed concurrently, there are no turns. There are also no dices, cards or any other random element in the game. The negotiation, the co-operative/competitive nature of the game and the absense of randomness make this game a very intertesting domain for testing MAS negotiation research. And make it a perfect target for our negotiation model.

The DipGame testbed facilitates the creation of player agents (bots) by providing a framework for bot development. It also provides the infrastructure and some software components that allow researchers to check and analyse the correct behaviour of their agents and an interface to chat with other players using the language $\mathcal{L}_1$. Detailed information about the testbed can be found in [8].

Multiagent systems' researchers can build negotiating agents with the DipGame framework and execute experiments with them using the available infrastructure, [7]. But what makes the testbed even more interesting is that also humans can play against DipGame bots throught the testbed website: `http://www.dipgame.org`. There is a huge community of Diplomacy players on Internet that meet around the `http://diplom.org` website. They publish a magazine, use software tools to study the games (like chess players do), write in fora available in several languages and participate in many regional and also international face-to-face competitions of

Diplomacy. Therefore, it is not difficult to find human players available for playing against our bots.

[8] discusses the complexity of the game and its adequacy to MAS research. There is a rather big community of bot developers, DAIDE, (over 200 members) and that build bots that are not capable of negotiation[4]. They share a common communication protocol and language that DipGame respects so bots build with the DipGame testbed will be able to compete against bots of the DAIDE community. The part of the communication in which DipGame departs from DAIDE is in the negotiation language and protocol[5].

10. DISCUSSION AND FUTURE WORK

In this paper we have outlined the basic components of a negotiation architecture. It is tailored to build agents capable to interact with humans in competitive environments. One of the main characteristics of the environments we envisage is that agents and humans can occasionally co-operate through joint plans in order to get a better outcome. Those plans are possible thanks to the negotiation processes ending successfully by signing deals where agents commit to perform actions.

Our main contributions are: first, the interleaving of on the fly generation of plans and options for the negotiation and the negotiation process itself; second, the use of a notion of coherence and diversity to better explore huge search spaces that allow for a more compact process of negotiation without erratic moves. We also advance the state of the art in the modelling of graded BDI [4] by better integrating BDI reasoning and planning.

We will build a *DipGame agent* using this architecture and test it playing against humans at the DipGame site. Our aim in these experiments is to empirically show that the architecture has the right components to be able to negotiate against humans in complex environments like the DipGame one: multialateral negotiations, repeated encounter scenarios, time fixed negotiation rounds, join tplans, uncertainty, ... Also, there are a number of extensions of the architecture that seem important when the interaction language among agents will be more complex. In this respect, our next step will be to extend the architecture allowing for the exchange of explicit information, using for instance language $\mathcal{L}_2$ [8].

Hypothetical reasoning in the sense of assuming dialogical moves and analysing their consequences might be included. It seems important that agents perform a forward thinking on the potential consequences of dialogical moves in order to plan the strategy ahead. Here we consider only one dialogical move ahead, and therefore the use of this reasoning will also be useful in future work.

The *emotional* dimension should be incorporated to the architecture as we want to interact with humans and they behave in a non totally rational way. They are deeply influenced by their emotions. Thus understanding the emotional state of the agents can help a lot in predicting their actions.

We evaluate a deal based on its value, that is, the expected utility of the plan. However, other measures could also be taken into account, e.g. social relationship building [12]. We think that the experiments that we will perform with humans will point us to the relevant measures. Learning techniques will be required to perform this task.

Finally, a very challenging aspect to look into is how to model *trustful image recovery* after a deceitful behaviour. That is, what strategy can we follow to be 'preceived as trustful again after performing a disloyal behaviour in which we did not honour a commitment. Agents lie and deceit to obtain a higher outcome. But we need to know, as human do, how to combine deceits with honour in a way that allows us to negotiate properly in the future.

Acknowledgements Research supported by the Agreement Technologies CONSOLIDER project under contract CSD2007-0022 and INGENIO 2010, by the Agreement Technologies COST Action, IC0801, and by the Generalitat de Catalunya under the grant 2009-SGR-1434.

11. REFERENCES

[1] An agent supports constructivist and ecological rationality. pages 255–258, Milano, 2009. IEEE Computer Society, IEEE Computer Society.

[2] J. Broersen, M. Dastani, J. Hulstijn, and L. van der Torre. Goal generation in the BOID architecture. *Cognitive Science Quarterly*, 2(3-4):428–447, 2002.

[3] V. c Torra and Y. Narukawa. *Modeling Decisions: Information Fusion and Aggregation Operators.* Springer, 2007.

[4] A. Casali, L. Godo, and C. Sierra. A logical framework to represent and reason about graded preferences and intentions. pages 27–37, Sydney, 2008. The AAAI Press, The AAAI Press.

[5] J. Debenham and C. Sierra. Trust and honour in information-based agency. pages 1225–1232, 2006.

[6] A. Fabregues, J. Madrenas, C. Sierra, and J. Debenham. Supplier performance in a digital ecosystem. In *DEST '09: Proceedings of the 3rd IEEE International Conference on Digital Ecosystems and Technologies*, Istanbul, Turkey, 2009.

[7] A. Fabregues and C. Sierra. Diplomacy game: the test bed. *PerAda Magazine, towards persuasive adaptation*, 2009. `http://www.perada-magazine.eu/view.php?source=1761-2009-08-03`.

[8] A. Fabregues and C. Sierra. A testbed for multiagent systems. Technical report, IIIA-CSIC, Bellaterra, Barcelona, 10/2009 2009.

[9] M. Georgeff, B. Pell, M. Pollack, M. Tambe, and M. Wooldridge. The belief-desire-intention model of agency. pages 1–10. Springer, 1998.

[10] R. Lin and S. Kraus. Designing automated agents capable of efficiently negotiating with people - overcoming the challenge. In *Proceedings of the 7th European Workshop on Multi-Agent Systems (EUMAS'09)*, 2009.

[11] Y. Shoham. AGENT0: A simple agent language and its interpreter. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, volume 2, pages 704–709, 1991.

[12] C. Sierra and J. Debenham. The logic negotiation model. In *Sixth International Joint Conference on Autonomous Agents and Multi-agent Systems*, pages 1026–1033, 2007.

[4]At least not about joint plans.

[5]DAIDE has defined a language for negotiation that we consider inappropriate. See [8] for a complete description on the reasons that made us define our own language (tower) for DipGame.