HANA: a Human-Aware Negotiation Architecture

Angela Fabregues and Carles Sierra

Institut d'Investigació en Intel·ligència Artificial, IIIA-CSIC,
Campus Universitat Autònoma de Barcelona, E-08193 Bellaterra, Barcelona, Spain.

Abstract

In this paper we propose HANA, a software architecture for agents that need to bilaterally negotiate joint plans of action in realistic scenarios. These negotiations may involve humans and are repeated along time. The architecture is based on a BDI model that represents the uncertainty on the environment as graded beliefs, desires and intentions. The architecture is modular and can easily be extended by incorporating different models (e.g. trust, intimacy, personality, normative, ...) that update the set of beliefs, desires or intentions. The architecture is dynamic as it monitors the environment and updates the beliefs accordingly. We introduce an innovative search&negotiation method that facilitates HANA agents to cope with huge spaces of joint plans. This method implements an anytime search algorithm that generates partial plans to feed the negotiation process. At the same time the negotiation guides the search towards joint plans that are more likely to be accepted.

Keywords: multiagent systems, automated negotiation, software architecture, practical reasoning, testbed, Diplomacy Game

1. Introduction

Since the development of software shifted towards networked systems in the mid 90s a lot of work has been done on automated bilateral negotiations [1, 2]. In most previous work autonomous software agents, usually selfish, interact using utility maximisation strategies [3, 4, 5]. These strategies usually work well when negotiation happens between software agents but not necessarily when humans are involved in the negotiations as recent work shows [6]. This is in part because humans do not follow a *constructivist* sort of

rationality [7, 8]. For instance, human decisions depend a lot on their social relationships [9], on emotions [10] and are contextualised in a particular culture [11].

Our long term research goal is to build software agents capable of negotiating with humans in complex real scenarios. More concretely, on how to negotiate joint plans of action among software agents and humans when bilateral negotiations can be intermingled. In this work we contribute to this agenda by formally describing the negotiation problem and by providing a concrete agent architecture. The architecture contains a number of elements that make it suitable for non-constructivist negotiations —by incorporating emotions, and apt for negotiating over large spaces of joint plans —by concurrently negotiating and searching for solutions. The architecture is inspired by an ecological type of rationality [12] as developed in the LOGIC theory of agency [9] and goes beyond it by proposing a concrete computational solution.

More concretely, we address in this paper the complex problem of simultaneous, repeated and bilateral negotiations in competitive environments. The agents are either software or human agents that compete but that can occasionally co-operate. The negotiation objects are joint plans of action. We are specially interested in negotiation domains that have a very large set of potential joint action plans as these are those with potential commercial interest (e.g. time tabling, team formation, supply chain management, gaming). In these scenarios, agents (and humans) need to negotiate joint action plans to improve their outcome. For instance, two teachers swapping time slots in their class schedules, or two members of a potential team negotiating the tasks to be performed. The environment is generally observable but the internal state of the other agents and their negotiations are usually private, that is, every agent can see the messages that it sends or receives but not the messages exchanged between any two other agents. In open systems, that is systems that allow unrestricted access of autonomous entities (either software agents or humans), reaching agreements on joint action plans is the way to figure out what our counterparts will do, and even this only to a certain extent as in some domains defection is possible. For instance, in Diplomacy, the case study used in this paper, a promise made by a player to perform a certain action may not materialise.

Negotiations are usually time framed. There is a deadline by which a negotiation process has to be finished. When these deadlines are tight, negotiators need to search quickly for good enough negotiation proposals instead

of looking for optimal proposals. For large solution spaces it is either not possible or too long to find them. This is, in fact, very common in humans' everyday life. Humans do not hesitate to take good enough decisions instead of waiting to be sure that the decision to take is the best one. Humans behave well in uncertain and competitive environments, as we are unsure of what the others will do and taking decisions quickly may be advantageous as the more time we wait the less opportunities to close a deal may be there. If an agent waits too long others may have reached agreements that are incompatible with the plans the agent likes.

The scenarios we are interested in witness repeated negotiations, for instance teachers negotiate every semester, or members of teams negotiate tasks for each problem to solve. These repeated interactions permit agents to build relationships, check whether the agreements are kept and act accordingly. If an agent breaks an agreement, it may become untrustworthy and the other agent involved in the agreement may penalise it, [13, 14]. A good way to penalise an agent is ignoring it, rejecting every proposal it makes as it makes little sense to reach agreements with someone that is untrustworthy: it will probably break the deal.

In summary, we address the problem of simultaneous bilateral negotiation of joint plans of action in competitive environments with repeated negotiation encounters and repeated rounds of plan execution. In these environments negotiation speed is crucial because as time goes by the number of available joint plans that can be accepted decreases.

The paper is structured as follows. We start providing a formal specification of the problem including the negotiation protocol and our case study in Section 2. Then, we introduce the agent architecture in Section 3 and describe its components in sections 4, 5 and 6. Finally, we conclude with a discussion and future work in Section 7.

2. Resource negotiation problems

In this section we formalise bilateral resource negotiation problems (RNP), that is, scenarios where agents negotiate about which actions to perform on the resources they control. The environment is dynamic, as it changes due to the uncontrollable actions of others. At each point in time its state contains a partition of the resources where each set of the partition corresponds to the resources controlled by a particular agent. The actions executed by agents

make the environment evolve. We model this evolution as a transition function between environment states. We assume without loss of generality that actions are performed synchronously at particular points in time. Also, we assume that negotiations between agents are iterative over a two-step process: (i) agents sign agreements on what actions to perform, and (ii) they execute the actions of the agreements. In the following two subsections we provide the formal specification of the environment and the negotiation protocol, and in the last subsection we introduce the game Diplomacy as an example of RNP. Diplomacy will be the case study used throughout the paper.

2.1. Environment

We consider environments that are fully observable and regulated by a set of rules (physical or otherwise) that determine their evolution. Environments are populated by agents $\mathcal{A}$ that control resources R and are always in one of several possible states.

Definition 1. *Given a set $\mathcal{A}$ of agents and a set R of resources, an* environment state *$\omega \subseteq \mathcal{A} \times R$ is a set of agent-resource pairs. We denote by W the set of all possible environment states, that is $W = 2^{\mathcal{A} \times R}$*

$\langle \alpha, r \rangle \in \omega$ means that agent α controls resource r and thus is the only agent that can act upon it.[1] We assume the existence of a finite set of operators Op that agents can apply to the resources they control. For instance, consuming the resource or transforming it. We thus define the set of possible actions as follows.

Definition 2. *The set of* actions *is the set $A = \mathcal{A} \times Op \times R$.*

We restrict the model to environments where no more than one operator can be applied to a resource simultaneously. This naturally leads to the definition of compatibility between actions.

Definition 3. *Two actions $a, b \in A$ such that $a = \langle \alpha, op_a, r_a \rangle$ and $b = \langle \beta, op_b, r_b \rangle$, are* compatible*, denoted by* $\mathrm{comp}(a, b)$*, if and only if $op_a = op_b$ implies $r_a \neq r_b$.*

[1] We will use the notation $\langle \cdot \rangle$ to denote elements of cartesian products.

Controlling a resource means that only the agent that controls the resource can act upon it. This is our notion of action feasibility.

Definition 4. *An action* $a = \langle \alpha, op, r \rangle \in A$ *is* feasible *in state* $\omega \in W$*, denoted by* $\text{feasible}(a, \omega)$*, if and only if* $\langle \alpha, r \rangle \in \omega$*.*

Actions are naturally grouped in sets, that we call plans, that without losing generality we can assume are executed at a given instant of time. Note that an agent can control more than one resource.

Definition 5. *A* plan $p \subseteq A$ *is a set of actions. The set of all possible plans is denoted by* $P = 2^A$*.*

We extend the notion of feasibility to plans in a natural way.

Definition 6. *Given a state* $\omega \in W$ *we say that plan* $p \in P$ *is feasible in state* ω*, denoted* $\text{feasible}(p, \omega)$*, if and only if for all* $a, b \in p$*,* $\text{feasible}(a, \omega)$ *and* $\text{comp}(a, b)$ *hold. The set of all feasible plans in state* ω *is denoted by* P^ω*.*

Two feasible plans are compatible if their actions are pair-wise compatible. That is, if its union is feasible.

Definition 7. *Given a state* $\omega \in W$ *and plans* $p, q \in P$*, we say that plans* p *and* q *are* compatible*, denoted* $\text{comp}(p, q)$*, if and only if their union is feasible, that is,* $\text{comp}(p, q) \Leftrightarrow p \cup q \in P^\omega$*.*

When an action is selected for each resource the plan is complete.

Definition 8. *Given a state* $\omega \in W$ *and a plan* $p \in P$*, we say that plan* p *is a* complete plan *for* ω *if and only if* $\text{feasible}(\omega, p)$ *holds and for all* $\langle \alpha, r \rangle \in \omega$ *then* $\langle \alpha, op, r \rangle \in p$ *for some* $op \in Op$*. We denote the set of all complete plans for state* ω *by* $\bar{P}^\omega \subseteq P^\omega$ *and by* $\bar{P}^\omega_\alpha$ *the projection of complete plans for* α*.*

Now we have all the ingredients to define environments as a type of deterministic transition system. That is, as a finite state machine with an initial state, with a set of final states, and with complete plans labeling the arcs between states.

Definition 9. *A* state transition system *is a tuple*

$$\Omega = \langle \mathcal{A}, R, Op, W, P, \mathbf{T}, \omega_0, W_f \rangle$$

where:

- $\mathcal{A}$ *is a set of agents*

- R *is a set of resources*

- Op *is a set of operators*

- $W = 2^{\mathcal{A} \times R}$ *is a set of states*

- $P = 2^{\mathcal{A} \times Op \times R}$ *is a set of plans*

- $\mathbf{T} : W \times P \to W$ *is a transition function such that* $\mathbf{T}(\omega, p)$ *is defined for all* $p \in \bar{P}^{\omega}$

- $\omega_0 \in W$ *is the initial state*

- $W_f \subseteq W$ *is the set of final states.*

The evolution of such a transition system is determined by a history of complete plans [2] being executed moving the state of the system away from the initial state and eventually reaching a final state.

Definition 10. *Given a transition system* $\Omega = \langle \mathcal{A}, R, Op, W, P, \mathbf{T}, \omega_0, W_f \rangle$, *a* history *is a sequence of complete plans* $H = \langle p_0, p_1, \ldots, p_n \rangle$ *such that for all* $0 < i < n$, $\mathbf{T}(\ldots(\mathbf{T}(\omega_0, p_0), \ldots), p_{i-1}) = \omega_i$ *and* $p_i \in \bar{P}^{\omega_i}$. *A history then implicitly defines an* environment state history *that is a sequence of states* $W_H = \langle \omega_0, \mathbf{T}(\omega_0, p_0), \mathbf{T}(\mathbf{T}(\omega_0, p_0), p_1), \ldots \rangle$ *that we refer to as* $W_H = \langle \omega_0, \omega_1, \omega_2, \ldots \rangle$.

2.2. Negotiation Protocol

We define the negotiation in an RNP to be bilateral and satisfy a particular protocol. As an environment contains many agents, multiple bilateral negotiations can take place even simultaneously. The set of plans over which two agents negotiate is the set of feasible plans containing just actions performed by them. The plans involving up to two agents are called *negotiation options*, or options to simplify.

[2] Notice that in some application domains, there are default actions that do not need to be explicitly executed by agents as they are assumed. For example, in Diplomacy units are assumed to hold as default. Or in time tabling, agents are assumed to continue with the current scheduling of its events. The default actions are part of the problem definition and thus they are know by all the agents.

Definition 11. *A feasible plan $\delta \in P^\omega$ is called a* negotiation option *if and only if*

$$0 < |\{\alpha \in \mathcal{A} | \langle \alpha, op, r \rangle \in \delta\}| \leq 2$$

We denote by $\mathcal{O}^\omega \subseteq P^\omega$ the set of negotiating options in state ω and by $\mathcal{O}^\omega_{\alpha,\beta} \subseteq P^\omega$ the negotiation options involving α and β.

When two agents enact a negotiation protocol, they propose options that involve the two agents. Agents alternate on the sending of proposals and accepting or rejecting them until time expires. The possible messages exchanged between two agents are called *negotiation utterances*.

Definition 12. *Given a transition system $\Omega = \langle \mathcal{A}, R, Op, W, P, \mathbf{T}, \omega_0, W_f \rangle$, a* negotiation utterance *in a state $\omega \in W$ is a tuple $\mu = \langle \theta, \alpha, \beta, \delta \rangle$ where $\theta \in \{\text{propose}, \text{accept}, \text{reject}\}$, $\alpha \in \mathcal{A}$ is the source and $\beta \in \mathcal{A}$ is the receiver of the utterance and $\delta \in \mathcal{O}^\omega_{\alpha,\beta}$. We denote by $M^\omega_{\alpha,\beta}$ the set of all possible negotiation utterances between α and β in state ω.*

In the following we indistinctively represent utterances as tuples or predicates, e.g. $\langle propose, \alpha, \beta, \delta \rangle \equiv propose(\alpha, \beta, \delta)$. We define negotiation dialogues as sequences of utterances sorted by time.

Definition 13. *Given a transition system Ω, a* negotiation dialogue Ψ *in state ω between α and β is a sequence $\Psi = \langle \mu_0, \mu_1, \ldots, \mu_n \rangle$ such that $\mu_i \in M^\omega_{\alpha,\beta}$ for all $0 \leq i \leq n$.*

The negotiation protocol illustrated in Figure 1 determines what can be said and in which order. Dialogues are formed as sequences of utterance so that each one is feasible with respect to the protocol. The next definition determines what utterances are feasible given a partial dialogue.

Definition 14. *Given a state $\omega \in W$, a dialogue $\Psi = \langle \mu_0, \mu_1, \ldots, \mu_{n-1} \rangle$ and an utterance $\mu_n = \langle \theta, \alpha, \beta, \delta \rangle$ we say that μ_n is* feasible *for dialogue Ψ, denoted by* $\text{feasible}(\Psi, \mu_n)$*, if and only if:*

- $\Psi = \langle\rangle \implies \theta = \text{propose}$
- $\mu_{n-1} = \langle \text{propose}, \beta, \alpha, \delta \rangle \implies \theta \neq \text{propose}$
- $\mu_{n-1} \neq \langle \text{propose}, \_, \_, \_ \rangle \implies \theta = \text{propose}$

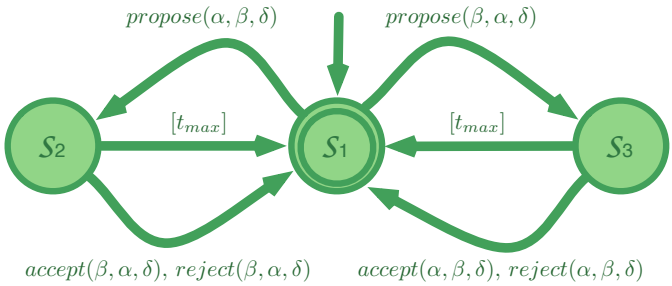


Figure 1: Negotiation protocol. Proposals are replied by accepts and rejects. It is not possible to send a proposal when a previous one is not yet replied. $[t_{max}]$ represents the end of the negotiation round.

The outcome of a successful negotiation that ended with an accept is a set of commitments on future actions to be performed by the negotiating agents. When an option being offered by agent α is accepted by agent β, it means that agents α and β commit to perform their actions in the option.

Definition 15. *Given a negotiation dialogue* $\Psi = \langle \mu_0, \mu_1, \ldots, \mu_n \rangle$ *we say that an action* $a \in A$ *is a* commitment *if* $\mu_i = \langle accept, \_, \_, \delta \rangle \in \Psi$ *and* $a \in \delta$. *We denote by* C^{Ψ} *the set of commitments in* Ψ.

2.3. Case study: Diplomacy

To illustrate the application of the agent architecture to a concrete case and to perform experiments we use the Diplomacy Game.[3] It is an example of the class of problems we intend to solve. It involves several players that repeatedly negotiate. All players perform their actions at the same time and the actions are made public so anyone can check whether its negotiating counterparts honoured the commitments reached at negotiation time. Its popularity,[4] the absence of random moves, the key role of negotiations during the game and the existence of literature on the strategy and tactics[5]

[3]The Diplomacy board game official website is `http://www.wizards.com/default.asp?x=ah/prod/diplomacy.Accessed25January2012.`

[4]Ranked 229 at `http://boardgamegeek.com/` by 5906 players.

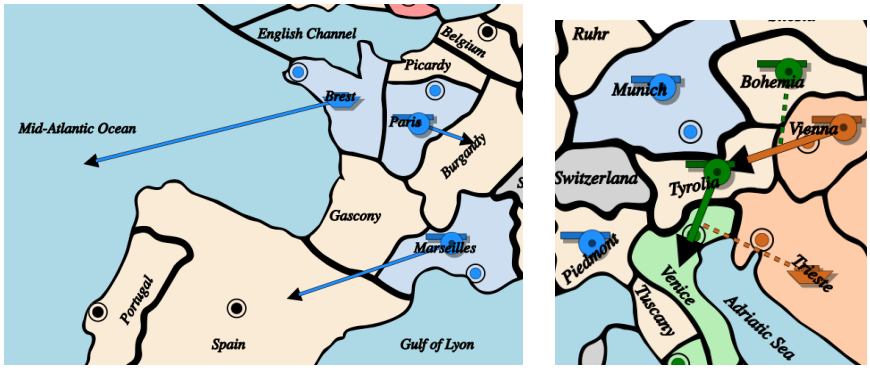
[5]The Diplomatic Pouch 'zine is the online magazine for the players of Diplomacy: `http://www.diplom.org/Zine/`. It contains articles analysing the strategy and tactics of the game.

make it the perfect case study to study concurrent bilateral negotiations. To perform experiments involving humans and software agents we have developed a testbed based on this game called DipGame that is freely available at `http://www.dipgame.org` [15, 16].

Diplomacy is played by seven players that control units spread over a map of Europe. Each player incarnates one of seven Great Powers: England, France, Italy, Germany, Austria, Russia and Turkey. The map is divided into several provinces that can be occupied by at most one unit. Players should strategically move their units over the map in order to build more units and thus eventually conquer Europe, which in practical terms means controlling at least eighteen provinces. The game spans several years and a year is composed of different phases. We focus on the *movement* phase of each year when players decide which movements should each one of their controlled units perform. The allowed movements of a unit are: (i) to hold, (ii) to move to an adjacent province, (iii) to hold and support a holding unit, and (iv) to hold supporting a moving unit. Supports are the only way to strengthen a unit that takes part in a battle. A battle happens when two (or more) units aim at being in the same province as a result of their movements (unit orders). As mentioned before, the announcement of orders is simultaneous. The movements over the board and the outcome of battles affect the creation of new units and thus determine the progress of players in the game. An in-depth description of the game can be found in [17].

Diplomacy is an RNP. The agents are the players (powers) and the units are the resources that they control. The operators that can be applied to resources are the different types of orders, that is, the name of unit movements according to the rules of the game (`http://en.wikibooks.org/wiki/Diplomacy/Rules`). Unit orders are the actions. During a movement phase the agents concurrently enact bilateral negotiations to reach agreements over the movements that they may jointly perform (e.g. getting the support of another player's unit to strengthen one of our units that will most probably get into battle in exchange of moving another of our units away from a certain province). At the end of a negotiation round all the agents announce their actions and as a consequence the board state changes.

The negotiation that takes place during the movement phases is usually performed in a natural language —English, for instance— as it is a game that is normally played by humans. The DipGame testbed contains a web application that allows people to talk to other people and to software agents by using of a formal language. The formal language is a standard of com-



(a) Plan (b) Joint plan

Figure 2: Two examples of plans.

munication among software agents using the testbed[6]. A standard is needed as software agents may be developed by different programmers. The testbed incorporates a translation library from (a reduced set of) English to the formal language and vice versa [15]. To illustrate the notion of plan we use the simplest of the formal languages and a simple ontology that only allows to represent orders [16].

In Figure 2 we graphically represent two plans, on 2a an individual plan for France and on 2b a joint plan between Italy and Austria.

The plan for France is to move from Paris to Burgandy, from Brest to Mid-Atlantic Ocean and from Marseilles to Spain. Thus the plan $p = \{a_1, a_2, a_3\}$ is a set of movements expressed in the formal language as:[7]

$$a_1 = \texttt{mto}(\texttt{Unit}(\texttt{fra}, \texttt{Region}(\texttt{par}, \texttt{amy})), \texttt{Region}(\texttt{bur}, \texttt{amy}))$$
$$a_2 = \texttt{mto}(\texttt{Unit}(\texttt{fra}, \texttt{Region}(\texttt{bre}, \texttt{flt})), \texttt{Region}(\texttt{mao}, \texttt{sea}))$$
$$a_3 = \texttt{mto}(\texttt{Unit}(\texttt{fra}, \texttt{Region}(\texttt{mar}, \texttt{amy})), \texttt{Region}(\texttt{spa}, \texttt{amy}))$$

In Diplomacy we cannot move the same unit to two different regions at the same time. This is an example of incompatibility of actions. Thus, given the action a_4 below, $\text{comp}(a_1, a_4)$ does not hold and therefore neither feasible(p) holds if p contains a_1 and a_4 nor $\text{comp}(\{a_1\}, \{a_2\})$ holds.

[6]The formal language is actually a language hierarchy with increasing complexity. For this paper we focus on the first level language that is the simplest of them.

[7]To simplify notation we represent the names of powers and provinces in a compressed way. For instance, France is "fra", Paris is "par" and Mid-Atlantic Ocean is "mao".

a_4 = mto(Unit(fra, Region(par, amy)), Region(gas, amy))

Assuming that Austria controls only two units, Figure 2b illustrates an example of joint plan for Austria $p = \{a_5, a_6, a_7, a_8\}$ where actions are represented as:

a_5 = mto(Unit(ita, Region(tyr, amy)), Region(ven, amy))
a_6 = sup(Unit(aus, Region(tri, flt)),
mto(Unit(ita, Region(tyr, amy)), Region(ven, amy)))
a_7 = mto(Unit(aus, Region(vie, amy)), Region(tyr, amy))
a_8 = sup(Unit(ita, Region(mar, amy)),
mto(Unit(aus, Region(vie, amy)),Region(tyr, amy)))

Given this plan, Italy could propose Austria different options:

- propose(ita, aus, [Commit(aus,ita, Do(a_6))]). Italy proposes Austria a deal where Austria commits to support the Italian move from Tyrol to Venice. As Italy does not give anything in exchange it may not be effective.
- propose(ita, aus, [Commit(ita,aus, Do(a_5)), Commit(aus,ita, Do(a_6))]). This looks more balanced but Italy is actually not providing anything beneficial to Austria.
- propose(ita, aus, [Commit(ita,aus, Do(a_5)), Commit(aus,ita, Do(a_6)), Commit(aus,ita, Do(a_7)), Commit(ita,aus, Do(a_8))]). This proposal is more balanced as Italy is also helping Austria.

Which one of these options should Italy actually send to Austria is what the architecture of a software agent and its negotiation strategy determine.

3. Agent architecture

In this section we propose a software architecture to build agents capable of participating in RNPs. We introduce here its main modules and then we give details for each of them in subsequent sections. We refer to the

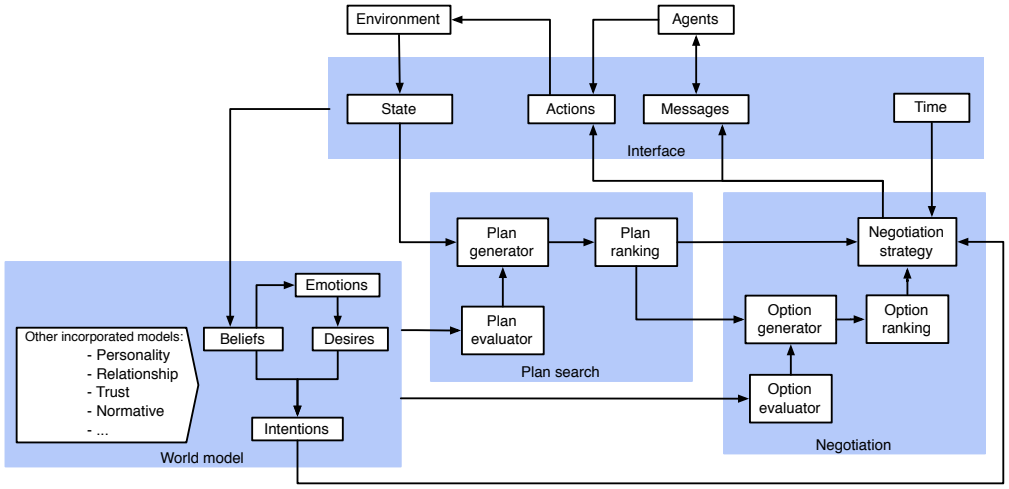


Figure 3: Graphical representation of HANA. Arrows represent data flows. Coloured boxes represent the modules that form part of the agent, and white boxes are components of those modules.

architecture as *HANA* and to the agents designed according to HANA as *HANA agents*.[8] The architecture is graphically represented in Figure 3.

In Section 2 we assumed that RNPs happen in fully observable environments that agents can perceive and where agents can execute actions whose (deterministic) consequences are also observable. Furthermore, the environment allows for agents to exchange messages. Thus, the first component of HANA is an *interface module* that situates the agents in their environment, that is, it allows to observe the environment state, observe and execute actions, and exchange messages with other agents. In other words, this module contains the sensors and actuators of the agent. Which actions to execute and which messages to send is decided by the *negotiation module*.

The design philosophy behind HANA is to provide some means to negotiate as humans do, as the negotiation counterparts could be humans. In particular, there are two capabilities that we think realistic agents should show: dealing with emotions and dealing with uncertainty, [18]. The architecture

[8]HANA is an acronym for Human-Aware Negotiation Architecture. We call the architecture this way to stress the aim of negotiating with humans in realistic scenarios that motivated the design of this agent architecture.

incorporates emotions as this is an important part of the non-constructivist rationality approach, we need to understand emotional reactions of the other negotiators. Although the environment is fully observable, the actions to be executed by the other agents can only be guessed analysing the other agents' previous behaviour. To cope with this uncertainty, we decided to represent the world as a graded BDI model, that is with graded *beliefs*, *desires* and *intentions* following the g-BDI model [19]. In Section 4 we provide a more in-depth description of the *world model module* that is the one containing the emotions and BDI components.

The space of plans and negotiation options that an agent can execute and propose, respectively, is potentially huge, as for example in The Diplomacy Game. Thus, we assume that the space is large enough and the negotiation time short enough to preclude obtaining the optimal. That means that any architecture for this type of negotiation needs to give the means to look for good enough solutions. Moreover, the longer it takes to decide what to propose the less probable it is the proposal to be accepted. As time goes by, the agents reach agreements that increase the amount of commitments and reduce the set of options compatible with the commitments. The increase of acquired commitments increases in turn the probability that our desired plans will not be compatible any longer. Consequently, the architecture must allow to start negotiating from the very beginning of a negotiation round. Dealing with huge solution spaces is not an inconvenient for human agents, e.g. in playing Chess or Go. Humans do work with good enough solutions in their everyday lives. Time constraints, boredom, or tiredness make humans accept good enough solutions. To start negotiating from the very beginning, HANA proposes to perform a search&negotiation method that assumes the plan search to go hand in hand with the negotiation. The *plan search module* executes an anytime algorithm that provides a periodically updated ranking of good enough plans. The ranking takes into account the commitments obtained by the *negotiation module*. And the negotiation module proposes options generated from the previously found good enough plans that contain actions to be executed by other agents. In this way, HANA agents can start the negotiation from the very beginning proposing options that, once negotiated, will provide new information —because the option will be accepted or rejected— to focus the search on the generation of new and better evaluated plans. As can be seen in Figure 3, the plan and option evaluators depend not only on the commitments but on the whole world module. Thus, those evaluation functions are also updated taking into account the intentions gen-

erated from new observations. The intentions trigger the decisions of the agent. In the following sections we provide a more in-depth description of the modules, their components and the decisions to be taken by the agent.

The execution of HANA consists of several concurrent processes for: the *interface* (to receive messages and observe the results of actions and the environment state), the *world model* (to update the world model given the perceived changes), the *plan search* (to continuously update the ranking of plans), and the *negotiation* (to generate options from plans and determine what to do next).

4. World model

For negotiation decisions to be effective they need to be based on an accurate assessment of the state of the environment and on the preferences of the other agents. Although the environment may be precisely known in certain domains, the preferences of others may be unknown, or may be uncertain. Moreover, specially in competitive scenarios, agents may lie about their preferences. This imposes the requirement that the world model has to be based on some uncertainty reasoning mechanism. During the last decade, some of the most successful representation models have been based on BDI representations. Most work on BDI models has concentrated on providing agent-oriented programming frameworks and languages such as Jason [20], Jack [21], AgentSpeak [22, 23], 3APL [24] or 2APL [25]; and on logical approaches to the BDI model such as modal logic [26], first-order logic [22], or belief degrees [27, 28]. BDI is based on the theory of human practical reasoning [29] and has well-founded logical semantics [30]. The work of Casali et al. [19] on what they denoted by *g-BDI*, gives a powerful generic representation for degrees of belief, degrees of desire (preferences) and degrees of intention (active goals). We adapt this work to our problem and incorporate the beliefs, desires and intentions as the main components of the world model.

We consider that desires and intentions are derived from beliefs. What happens in the world determines whether we desire it to change into a different world and whether we intend to make that happen. In this section we concentrate on how HANA agents represent their beliefs about the next environment state. The most important aspect of the uncertain evolution of the world is what an agent expects to happen in the environment due to the decisions of other agents. This is so because the natural evolution

of environments is subject to shared knowledge on physical laws and thus known by every agent. Therefore, the evolution of the world can be due to either actions, A, or utterances, M, of other agents. We denote those events (actions and utterances) by $\Phi = M \cup A$. The agent has at all time a belief degree assigned to every element of Φ meaning how certain is the agent that that event will happen. We decided to model these degrees as probabilities because the data for them comes from the previous interactions with the other agents and thus those data can be statistically processed. Axiomatics on how to represent probabilities are provided in [19].

Definition 16. *Given $\Phi = M \cup A$, a* belief *is a tuple $\langle \alpha, \varphi, \vartheta \rangle \in \mathcal{A} \times \Phi \times [0,1]$ where ϑ is α's belief degree on φ happening. We denote by $\mathcal{B}$ the set of all possible beliefs and by $\mathcal{B}_\alpha \subseteq \mathcal{B}$ the set of possible beliefs of α.*[9]

We define the feasibility of $\varphi \in \Phi$ as an extension of the feasibility on utterances and actions.

Definition 17. *Given the current state $\omega \in W$ and the current dialogue Ψ, we define feasibility for $\varphi \in \Phi$ as:*

$$\text{feasible}(\varphi) = \begin{cases} \text{feasible}(\omega, \varphi) & \textit{if} \quad \varphi \in A \\ \text{feasible}(\Psi, \varphi) & \textit{if} \quad \varphi \in M \end{cases}$$

The particular type of problem studied here gives some restrictions on over what it is feasible to happen. All agents know the physical laws of the world, thus they all know that non feasible events will not happen. The degree of belief on a non feasible event would be then the minimum, 0. Also for plans: non feasible plans will not happen as those plans contain non feasible or non compatible actions. In the following we show two constraints for values of the belief model that are derived from the notion of feasibility and compatibility. In particular, only one operator can be applied to a resource. Two feasible actions operating on the same resource are non compatible and, thus, they cannot both happen at the same time. As we decided to represent degrees as probabilities, the summation of probabilities on all those actions to be operated on the same resource must be 1:

$$\forall \omega . \forall \langle \alpha, r \rangle \in \omega . \sum_{\{a_i = \langle \alpha, op_i, r \rangle | \text{feasible}(a_i, \omega) \text{ and } op_i \in Op\}} B_\alpha(a_i) = 1 \qquad (1)$$

[9] We will note $\langle \alpha, \varphi, \vartheta \rangle \in \mathcal{B}$ as $B_\alpha(\varphi) = \vartheta$ when useful.

Moreover, the negotiation protocol proposed in this work imposes that only feasible utterances are possible.

$$\forall \Psi. \sum_{\{\mu_i | \text{feasible}(\Psi, \mu_i)\}} B_\alpha(\mu_i) = 1 \tag{2}$$

For a given environment state ω, the belief degree of α on an action a happening is $B_\alpha(a)$. Recall that plans are considered sets of actions that are to be executed at the same time. The belief on the execution of a feasible plan $p = \{a_1, a_2, \ldots, a_n\} \in P^\omega$ is thus naturally modelled according to HANA as the belief on the conjunction of the execution of each action that is then modelled as the product.

$$B_\alpha(p) = B_\alpha(a_1 \wedge a_2 \wedge \cdots \wedge a_n) = \Pi_{a_i \in p} B_\alpha(a_i) \tag{3}$$

When new observations of the environment are made, HANA agents update their beliefs. From the many possible *belief review functions* available in the literature [31, 32, 28] HANA uses a recency based belief revision defined as follows.

Definition 18. *Given an agent $\alpha \in \mathcal{A}$ a* belief review function*, denoted by $\sigma : 2^{\mathcal{B}_\alpha} \times 2^{\mathcal{B}_\alpha} \to 2^{\mathcal{B}_\alpha}$, is any function satisfying:*

- $\sigma(\mathcal{B}', \mathcal{B}'') = \mathcal{B}'''$
- $\mathcal{B}'$ *is the original belief set*
- $\mathcal{B}''$ *is a new belief set*
- *If* $\langle \varphi, \vartheta \rangle \in \mathcal{B}'$, $\langle \varphi, \vartheta' \rangle \in \mathcal{B}''$ *and* $\vartheta \neq \vartheta'$ *then* $\langle \varphi, \vartheta' \rangle \in \mathcal{B}^*$
- *If* $\langle \varphi, \vartheta \rangle \in \mathcal{B}'$ *and* $\langle \varphi, \_ \rangle \notin \mathcal{B}''$ *then* $\langle \varphi, \vartheta \rangle \in \mathcal{B}^*$
- *Nothing else belongs to* $\mathcal{B}^*$
- $\mathcal{B}'''$ *is the normalization*[10] *of* $\mathcal{B}^*$

[10] To normalise, we follow the work done in [33] and compute the minimum relative entropy probability distributions with respect to the distributions in $\mathcal{B}'$ that consider the new beliefs in $\mathcal{B}''$ as constraints to satisfy and that satisfy equations 1 and 2.

Desires, intentions and emotions[11] are also updated via a similar update method that for brevity we omit here. The arrows in Figure 3 show the influence between the different motives: changes in the environment provoke updates in the belief set that generate updates in the emotions that update the desires. The new set of beliefs and desires determines new intentions.

The world model architecture allows to represent complex behaviours as next example show.

Example 1. *The HANA agent* Revenger *plays Diplomacy and is configured to have a sensitive and aggressive personality. It comes to believe that the agent* Beth *is an enemy as it executed a movement that attacks one of Revenger's units and never accepted any proposal in the past three negotiation rounds. Its personality rules for revenge trigger a desire (with a high degree) to damage Beth that will in turn give an intention (again with high degree) to attack one of Beth's units although it is a very difficult task and there are other alternative plans that would be easier to reach and would give Revenger a higher rational utility.*

Although only a few components in Figure 3 are interconnected to build up the world model of HANA agents (beliefs, desires, intentions and emotions), other models might be incorporated. The world model is based on multicontext systems [19] that are modular structures that allow for an easy interconnection of different (logical) models, using transition functions between them, to build even more complex agents. For instance, a trust model may impact on intentions as the intention degree to satisfy a desire via a plan with an untrustworthy agent should be low. Also, a social model might impact on intentions as we might want to have a higher intention degree on plans involving an agent with whom we would like to increase the level of intimacy [9] than otherwise.

As defined in Section 2.2, the agents must fulfil a negotiation protocol in order to be able to negotiate with other agents. The rules or constraints that the protocol provides can be incorporated in the agent as internal norms to

[11] To model emotions we follow [34] and represent *emotional states* or *moods*. The personality of the agent is part of the emotional model and determines how to update the mood of the agent according to the beliefs about the environment. Personality may connect for instance frustration (a belief on a failed negotiation) with retaliation (a desire of a non-rational future negotiation).

follow. This is done according to HANA with a high desire degree on fulfilling the negotiation protocol and some transition functions between this desire and several beliefs that modify the degree of what we call *basic intentions*: *reply* δ (when the agent beliefs that it received proposal δ), *propose* (when there is time left in the negotiation round) and *executeActions* (when we are approaching the end of the negotiation round).

5. Plan search

The interplay between search and negotiation is the most important contribution of HANA. In most multi-issue negotiation problems the space of potential solutions is determined by the admissible values of issues, that is, potential solutions are elements of the cartesian product of the admissible values for each issue [35]. Differently, in RNPs the space of potential solutions is defined as the combination of feasible actions that are compatible. Only certain subsets of the space of actions constitute feasible solutions and finding which ones are feasible is not straightforward. Good and bad solutions are not placed together nicely as in continuously valued attributes (e.g. if a certain low price is good, nearby low prices will be similarly good). Sometimes a small change in a plan makes it go from excellent to catastrophic. Moreover, the space of potential solutions in real domains is frequently huge. What HANA brings in to address in this type of negotiation problem is a search&negotiation method that enables the negotiation to start as soon as possible over reasonably good solutions. We explain the details of how solutions are sought in this section.

The outcome of the search process is a continuously refreshed ranking of candidate plans. The *plan ranking* is made by the *plan generator* thanks to a utility function that represents the preferences of the agent and that is implemented within a component of the architecture called *plan evaluator*, see Figure 3. Preferences are determined by the world model and thus take into account personality traits or relationships between agents for instance, not just an individual position improvement.

Every agent α in a state transition system Ω must decide what actions to perform, that is, what complete plan $\bar{p}_\alpha$ to perform. Remember that given a state ω, next state ω' is computed by a state transition function $\mathbf{T} : W \times P \to W$. $\mathbf{T}(\omega, \bar{p})$ is defined for complete plans, $\bar{p} \in \bar{P}^\omega$, that are those that can be obtained from the union of complete plans for every agent controlling resources in the current state: $\bar{p} = \bigcup_{\beta \in \mathcal{A}} \bar{p}_\beta$. To decide what plan

to perform, α must know what is the utility that every plan would provide. If α knew the plans of the other agents, $Q = \bigcup_{\beta \in \mathcal{A} \setminus \{\alpha\}} \bar{p}_\beta$, it could compute the utility of α performing the complete plan $\bar{p}_\alpha$ using its utility function, $\mathcal{U}_\alpha : W \to [0, 1]$, as:

$$\mathcal{U}_\alpha(\omega, \bar{p}_\alpha) = \mathcal{U}_\alpha(\mathbf{T}(\omega, Q \cup \bar{p}_\alpha))$$

However, whilst agents may have a clear idea of their preferences and hence can build up a utility function, it is usually impossible to know what the plans of other agents will be. Therefore, instead of using the deterministic transition function, $\mathbf{T} : W \times P \to W$, α has to use a probabilistic state transition function.

Definition 19. *Given a transition system* $\Omega = \langle \mathcal{A}, R, Op, W, P, \mathbf{T}, \omega_0, W_f \rangle$, *a* probabilistic transition function, *denoted by* $\mathbb{T}(\omega' | \omega, p) \in \mathbb{P}(W)$ *is any conditional probability distribution over* W *given* $\omega \in W$ *and* $p \in P$, *such that for every complete plan* $\bar{p} \in \bar{P}^\omega$ *then* $\mathbb{T}(\mathbf{T}(\omega, \bar{p}) | \omega, \bar{p}) = 1$ *and* $\mathbb{T}(\omega' | \omega, \bar{p}) = 0$ *for all* $\omega' \neq \mathbf{T}(\omega, \bar{p})$.

In RNPs the state transition function $\mathbf{T} : W \times P \to W$ is fixed and common to all the agents. Instead, a probabilistic transition function has to be particular for each agent as it necessarily depends on the interpretation of the past behaviours of other agents. Therefore, we will denote by $\mathbb{T}_\alpha(\omega' | \omega, p)$ the probabilistic transition function of agent α. Then, the utility of a plan, complete or not, for an agent can be estimated as follows:

$$E[\mathcal{U}_\alpha(\omega, p)] = \sum_{\omega_i \in W} \mathbb{T}_\alpha(\omega_i | \omega, p) \times \mathcal{U}_\alpha(\omega_i) \tag{4}$$

The complexity here relies on the evaluation of $\mathbb{T}_\alpha(\omega' | \omega, p)$.

We can identify the problem of learning the probabilistic state transition function for all complete plans for a given agent $\alpha \in \mathcal{A}$ as a Markov Decision Process (MDP), [36]. A MDP is a tuple $\langle S, A, L : A \times S \times S \to [0, 1], R : A \times S \times S \to [0, 1] \rangle$, where S is a finite set of states, A is a finite set of actions, $L(a, s, s')$ is the probability that action a in state s will lead to state s', and $R(a, s, s')$ is the immediate reward received after transition to state s' from state s with transition probability $L(a, s, s')$. The interpretation as an MDP is based on having $S = W$, $A = P$, $L = \mathbb{T}$ and $R(p, \omega, \omega') = \mathcal{U}_\alpha(\omega') - \mathcal{U}_\alpha(\omega)$. Learning $\mathbb{T}$ requires a wealth of data that is usually not available

and an initially random behaviour that may produce very negative outcomes in RNPs. Moreover, there is a required feature for any MDP problem that is not verified in our case: the Markov property. The transition function could depend on the past as other agents could learn from previous states and modify their decision function.

We propose an alternative to model the problem as an MDP that is to infer the probability state transition function from beliefs on the execution of plans as defined in Section 4. From belief degrees on particular actions happening we can compute the belief degree of complete plans. Also, beliefs easily integrate other sources of information that are missing in a MDP, such as emotions or previous commitments. That is, the belief degree on an action happening may be determined, for instance, by knowing that the other agent is of a revenge type or that the other agent reached an agreement with another agent whom we trust and told us so. Moreover, as new sources of information can be easily incorporated into the world model this makes the architecture highly flexible and modular. For all those reasons we define the expected utility not for a plan in particular but for a set of beliefs hold in a particular state as follows:

Definition 20. *We define the* expected utility *for $\alpha \in \mathcal{A}$ holding the set of beliefs $\mathcal{B}' \subseteq \mathcal{B}_\alpha$ in state $\omega \in W$ as:*

$$E[\mathcal{U}_\alpha(\omega, \mathcal{B}')] = \sum_{\omega_i \in W} \left(\frac{\sum_{\substack{\bar{p} \in \bar{P}^\omega \\ \mathbf{T}(\omega, \bar{p}) = \omega_i}} B'(\bar{p})}{\sum_{\bar{p} \in \bar{P}^\omega} B'(\bar{p})} \times \mathcal{U}_\alpha(\omega_i) \right) \tag{5}$$

The previous definition does not require that α has made up his decision of what actions to perform. That is, the equation can be used at the beginning of the negotiation process —when α is still uncertain on what to do, and when all the bi-lateral negotiation processes have been finished and α knows what to do. The expected utility of a plan p is computed at any time assuming that the plan will be executed: $E[\mathcal{U}_\alpha(\omega, \sigma(\mathcal{B}', \{\langle \alpha, a, 1\rangle | a \in p\}))]$[12]. Actually, the richer the world model the more accurate the utility functions can be. We measure the level of information in a belief set as the average of Shannon's entropy among the probability distributions of actions to be operated on

[12]Plans are executed at the end of the negotiation round when the certainty on the actions to be executed use to be high.

resources. Notice that the higher the uncertainty the higher the entropy and thus, the less information.

Definition 21. *The uncertainty on a set of beliefs $\mathcal{B}' \subseteq \mathcal{B}_\alpha$ given the set of predicates Φ that partition it, is measured as:*

$$\mathcal{H}(\mathcal{B}') = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} \left(\frac{1}{\ln |\phi|} \sum_{\langle \alpha, \varphi_i, \vartheta_i \rangle \in \phi} \vartheta_i \ln \vartheta_i \right) \tag{6}$$

The uncertainty on actions to be executed is usually high at the beginning of a negotiation round, as an agent does not have enough information about each agent decisions, to 0 when the complete plan is actually performed and observed by all agents. Negotiation is the means to reduce the uncertainty on the belief model. By reaching agreements on negotiation options agents commit to the execution of their actions in the negotiating option and thus reduce the uncertainty by making equal to zero the probability of executing incompatible actions on the same resource.[13]

The task of the plan search module is to find good enough plans to be executed by the agent, but also to provide good enough plans to negotiate with other agents. Plans to be executed are complete plans for the agent, that is, plans containing actions involving all the resources controlled by the agent in the current environment state. The plans to negotiate with other agents are extensions of those complete plans containing actions to be performed by other agents.

Definition 22. *Given a transition system $\Omega = \langle \mathcal{A}, R, Op, W, P, \mathbf{T}, \omega_0, W_f \rangle$ and the current state $\omega \in W$, a* joint plan *in state ω is a plan $p \in P^\omega$ involving at least two agents, $|\{\alpha | \langle \alpha, op, r \rangle \in p\}| \geq 2$, and complete for one of them, that is, there is $\bar{p} \in \cup_{\alpha \in \mathcal{A}} \bar{P}^\omega_\alpha$ such that $\bar{p} \subseteq p$ We denote the set of joint plans in ω by $\hat{P}^\omega$ and the set of joint plans with a complete plan for α by $\hat{P}^\omega_\alpha$.*

The bilateral nature of the interactions force options to contain actions to be done by, at least, two agents. In HANA, joint plans involving more than

[13] If a trust model is used and the trust in the opponent is not complete, then the belief on that agent executing an action incompatible with its commitment may not be zero. Similarly, low trust values on an agent increase the intention of not negotiating with it.

two agents can be negotiated (either concurrently or sequentially) proposing several options generated from the joint plan. In Section 6 we explain how options are generated from joint plans. In this section we focus on the plan search. As introduced before, plans can be evaluated by their expected utility. Notice that this measure assumes that the plan will be executed. When joint plans are evaluated, we can also assume that the HANA agent will execute its part of the plan. Even though, we must be cautious about the actual execution of the actions in the plan assigned to other agents. We define the *confidence* of a plan in order to measure the degree of belief on the execution of a joint plan assuming that the HANA agent will perform its part of the plan.

Definition 23. *Given a set of beliefs $\mathcal{B}' \subset \mathcal{B}_\alpha$ hold by agent $\alpha \in \mathcal{A}$, and a feasible plan $p \in P^\omega$, α's* confidence *on p is defined as:*

$$\mathcal{C}_\alpha(\mathcal{B}', p) = B(\{\langle \beta, op, r\rangle | \langle \beta, op, r\rangle \in p \text{ and } \beta \neq \alpha\})$$

Note that for all $\bar{p}_\alpha \in \bar{P}^\omega_\alpha$, $\mathcal{C}_\alpha(\mathcal{B}', \bar{p}_\alpha) = 1$.[14]

The output of the plan search is a *plan ranking*. The confidence measure can be used by HANA agents to rank joint plans. HANA agents can rank the joint plans by their utility (how good are they for me) filtering out those joint plans that do not reach a minimum level of confidence (the agent does not think that other agents will perform their actions in the plan). This minimum level of confidence may increase as time goes by and the negotiation round deadline approximates in order to focus on joint plans which confidence is high.

To generate plans we need a search algorithm that is: (i) capable to search in a huge space of solutions —as required by most real scenarios, (ii) anytime —as required by the time bounds, (iii) capable to generate several solutions instead of just one —we are looking for several plans, and (iv) guided by a dynamic heuristic —as the set of beliefs evolves with the agent interaction. We decided to implement HANA's plan generator with an evolutionary algorithm that constantly optimise the set of plans in the ranking. Concretely we use a genetic algorithm (GA) as these algorithms allow to efficiently explore large spaces of solutions and produce successive populations of solutions that are increasingly better adapted to the environment even when it is changing along the search process. For us, each single solution, a chromosome in

[14]This is so because $B(\emptyset) = B(true) = 1$.

the GA, represents a complete or joint plan for the agent. The idea is to generate the plan ranking from the current population of solutions taking all or a subset of the best ones, preferably the latter. This population is updated generation after generation by the crossover, mutation and selection operators. It is important to guarantee the feasibility of the generated plans when applying crossover and mutation. The evaluation of a chromosome is done by the fitness function that computes the expected utility of the plan represented by the chromosome. *Fitness proportionate selection* is used to give more chances to good plans to take part of crossovers. HANA's genetic search allows to set the probability of genes being mutated. We use this to focus the search on the joint plans looking for other agent actions that can nicely extend the best complete plans. In fact, to represent plans of diverse size, we use chromosomes with size equal to the size of complete plans and let some genes to have a void value meaning that the resource corresponding to that gene has no assigned action. The probability of void values per gene can also be adjusted. The initial population does not need to be randomised. We set the initial population and stop the search at any time saving the current population. In this way, we can resume the computation later on. It is possible to use elitism to keep the best plans alive generation after generation. Elitism in the plan generation provides a minimum of stability needed to avoid an erratic performance of the agent during negotiation. The idea is to keep the plan search running all the time but it can be stopped and restarted if it is necessary.

Concluding, in this section we have argued the need of using data from the world model to evaluate plans. We described how plans are evaluated and how the evaluation functions change as new observations update the world model usually reducing the agent's uncertainty. And finally, we explained the necessary features that a plan generation should have and how we implemented the plan generation in HANA agents to generate the ranking of plans.

6. Negotiation

The negotiation module uses the ranking of plans and the world model to decide how to negotiate and what actions to perform. That is, what messages to send to other agents and what actions to execute over the environment. The world model and the plan search have independent processes that make the world model data and the plan ranking evolve along time.

The negotiation module is controlled by another process that periodically takes a snapshot of both previous processes' data structures. We define this snapshot as a *negotiation state*.

Definition 24. *A* negotiation state *is a tuple*

$$s = \langle \alpha, \omega, t, \mathcal{B}^t_\alpha, \mathcal{D}^t_\alpha, \mathcal{I}^t_\alpha, P^t_\alpha \rangle$$

where:

- $\alpha \in \mathcal{A}$ *is an agent*
- $\omega \in W$ *is an environment state*
- t *is a time instant*
- $\mathcal{B}^t_\alpha$, $\mathcal{D}^t_\alpha$, *and* $\mathcal{I}^t_\alpha$ *are the beliefs, desires and intentions of* α *at time* t
- $P^t_\alpha : P^\omega \mapsto [0, 1]$ *is a plan ranking*

We denote the set of all possible negotiation states by S. Taking a snapshot the negotiation process can use the data from the world model and the plan ranking and perform a negotiation step while the plan search is looking for even better plans[15]. The workflow of the negotiation process is as follows:

1. Takes a snapshot of the current negotiation state.
2. Generates a ranking of negotiation options.
3. Executes the agent's intentions included in the world model.
4. Goes to (1) to continue with a new negotiation state.

The negotiation process uses the option generator to build a ranking of negotiating options satisfying the Definition 11. Options are generated from the plan ranking P^t_α as combinations of actions in joint plans $\hat{p} \in \hat{P}^\omega_\alpha$ that are included in the plan ranking $P^t_\alpha(\hat{p}) \neq \bot$. Options are evaluated by the option evaluator that computes the next expected negotiation state assuming the acceptance of a given option $\delta \in \mathcal{O}^\omega_\alpha$. The simplest way to generate the ranking of options, $\mathcal{O}^t_\alpha : \mathcal{O}^\omega_\alpha \mapsto [0, 1]$ is as follows:

[15]Note that the changes in the world model and the plan ranking that are done after taking the snapshot are considered in the next iteration of the negotiation process.

$$\mathcal{O}_\alpha^t(\delta) = f(next(s, \delta))$$

where $s \in S$ is the current negotiation state, $f : S \mapsto [0, 1]$ is a negotiation state evaluation function, $next : S \times \mathcal{O}_\alpha^\omega \mapsto S$ computes the next expected negotiation state, and $\exists \hat{p} \in \hat{P}_\alpha^\omega$ such that $\delta \subseteq \hat{p}$ and $P_\alpha^t(\hat{p}) \neq \bot$.

An alternative is to apply a filter and generate the option ranking using only the joint plans with value over a threshold $\nu > 0$. That is, using every plan $\hat{p} \in \hat{P}_\alpha^\omega$ such that $P_\alpha^t(\hat{p}) > \nu$.

Given the negotiation state $s = \langle \alpha, \omega, t, \mathcal{B}_\alpha^t, \mathcal{D}_\alpha^t, \mathcal{I}_\alpha^t, P_\alpha^t \rangle$ and the option δ, an example of next expected negotiation state $next(s, \delta)$, for a HANA agent that always honours its commitments and fully trust the other agents, could be the negotiation state $s' = \langle \alpha, \omega, t', \mathcal{B}_\alpha^{t'}, \mathcal{D}_\alpha^{t'}, \mathcal{I}_\alpha^{t'}, P_\alpha^{t'} \rangle$ such that $\mathcal{B}_\alpha^{t'} = \sigma(\mathcal{B}_\alpha^t, \{\langle \alpha, a, 1 \rangle | a \in \delta\})$ and $P_\alpha^{t'} = \{p | p \in P_\alpha^t \text{ and } \text{comp}(p, \delta)\}$. The desires and intentions would be updated according to this new set of beliefs.

HANA provides several evaluation functions for negotiation states. Other functions can be used. In general, the reacher the world model the more sophisticate the evaluation functions can be. The following functions assume the basic world model with beliefs on actions.

- *Quality of information.* The higher the quality of the information that we can reach in a negotiation state the better. A well informed state contains joint plans that can reduce the uncertainty about the other agents' actions. The higher the uncertainty reduction the better. A natural way to evaluate the quality of information is to define it as 1 minus the average uncertainty of Definition 21.

$$f_H(s) = \max_{p \in P_\alpha^t} (1 - \mathcal{H}(\sigma(\mathcal{B}_\alpha^t, \{\langle \alpha, a, 1 \rangle | a \in p\}))) \tag{7}$$

- *Independence.* The more independent an agent is the better. If an agent can reach a high utility by its own means the better the negotiation state is. This measure depends on the complete plans for the agent that have been found so far, $\bar{P}_\alpha^t = \{p | p \in \bar{P}_\alpha \text{ and } P_\alpha^t(p) \neq \bot\}$. A state is as good as the best state the agent can reach by its own means, this is the maximum expected utility to be obtained by assuming we chose one of the complete plans for agent α:

$$f_{\mathcal{U}C}(s) = \max_{\bar{p} \in \bar{P}_\alpha^t} E[\mathcal{U}_\alpha(\omega, \sigma(\mathcal{B}_\alpha^t, \{\langle \alpha, a, 1 \rangle | a \in \bar{p}\}))] \tag{8}$$

- *Opportunity.* The more utility to be obtained with joint plans the better. Finding joint plans that give high utility is actually the reason of the whole negotiation process. Any state that has joint plans with high expected utility is a good state. This measure is similar to the previous one but using joint plans $\hat{P}_\alpha^t = \{p | p \in \hat{P}_\alpha \text{ and } P_\alpha^t(p) \neq \bot\}$:

$$f_{\mathcal{U}J}(s) = \max_{\hat{p} \in \hat{P}_\alpha^t} E[\mathcal{U}_\alpha(\omega, \sigma(\mathcal{B}_\alpha^t, \{\langle \alpha, a, 1 \rangle | a \in \hat{p}\}))] \tag{9}$$

- *Confidence.* The more confidence in the available plans the better. Having a high confidence in the plans found during the search the less uncertainty on what will happen.

$$f_{\mathcal{C}}(s) = \max_{p \in P_\alpha^t} \{\mathcal{C}_\alpha(\mathcal{B}_\alpha^t, p) | p \in P^\omega, P_\alpha^t(p) \neq \bot\} \tag{10}$$

Each of these different measures, or a combination of them, allows to evaluate negotiation states and thus rank the available options. When to use each measure or how to combine them is what determines an agent's negotiation strategy. HANA allows to define strategies combining these measures. The other key element of the negotiation strategy is the *aspiration level*, i.e. the minimum evaluation value, that the agent has for options to be acceptable. The options above the aspiration level should be accepted. Otherwise, rejected. At the beginning of a negotiation round agents would usually request a high aspiration value. As time goes by and the deadline approaches, agents become less demanding and decrease their expectations in order to reach some agreements that improve, even in a low amount, their negotiation state. HANA allows to define the way the aspiration level decreases as the next definition shows.

Definition 25. *Given a negotiation state s, a deadline t_{max}, and current time t, the* aspiration level*, denoted $\mathcal{A}(s,t)$, is defined as:*

$$\mathcal{A}(s,t) = g_{min}(s,t) + \left(\frac{t_{max} - t}{t_{max}}\right)^\tau \cdot (1 - g_{min}(s,t))$$

where $\tau \in [0,1]$ is the aspiration decay rate and $g_{min}(s,t)$ is the minimum value that can be guaranteed.

The negotiation strategy is then determined by fixing values for $g_{min}(s,t)$. HANA allows to define these functions as linear combinations of the measures defined before. That is,

$$g_{min}(s,t) = w_1(t) \cdot g_1(s) + w_2(t) \cdot g_2(s) + ... + w_n(t) \cdot g_n(s)$$

where every $g_i(s) \in [0,1]$ is a measure over the state s and $\sum_{i<n} w_i(t) = 1$. Next we discuss a few negotiation strategies:

- *Conservative*. An agent can guarantee a minimum utilitarian value with its own actions that corresponds to $g_{min}(s) = f_{\mathcal{UC}}(s)$. This strategy is convenient at the end of a negotiation round as it concedes maximally towards the guaranteed minimum.

- *Informative*. A convenient strategy at the beginning of a negotiation round is to increase the agent's information quality. This facilitates to explore the space of options and reduce uncertainty in the negotiation. The more information an agent has the more probable its future proposals will be accepted. This can be achieved by $g_{min}(s) = f_H(s)$.

- *Dynamic*. A combination of the previous two starting with informative and ending with conservative.

  $$g_{min}(s,t) = w_1(t) \cdot f_H(s) + w_2(t) \cdot f_{\mathcal{UC}}(s)$$

  where $w_1(t) = \frac{t_{max}-t}{t_{max}}$ and $w_2(t) = 1 - w_1(t)$.

As introduced in Section 4, the HANA agents have three basic intentions that are: *reply* δ, *propose* and *executeActions*. Desires and intentions are graded and are represented similarly to how beliefs are represented. The basic intentions are mutually incompatible, thus the aggregation of their degrees is always 1. HANA agents satisfy the negotiation protocol defined in Section 2 providing transition functions from beliefs and desires to intentions. Those transition functions update the degrees of the basic intentions whenever a message is sent or the deadline is reached. The negotiation process executes the current basic intention with the highest degree.

At the beginning of the negotiation round, the highest basic intention is always to *propose*: $\mathcal{I}^t_\alpha(\text{propose}) > \mathcal{I}^t_\alpha(\text{reply } \delta) \wedge \mathcal{I}^t_\alpha(\text{propose}) > \mathcal{I}^t_\alpha(\text{executeActions})$. That is also the case when there is no proposal received and there is time left before the timeout. Whenever the highest basic intention is *propose*, the negotiation process executes the following sentences proposing the best option only when it is better than the current aspiration of the agent:

Ensure: s is the current negotiation state
  $\delta \leftarrow \arg\max_{\delta \in \mathcal{O}_\alpha^\omega} \mathcal{O}_\alpha^t(\delta)$ {selects the best ranked option}
  if $\mathcal{O}_\alpha^t(\delta) > \mathcal{A}(s,t)$ **then**
    propose(δ) {the interface module sends a message proposing δ}
  end if

When a proposal is received, the highest basic intention becomes to be *reply* δ that is to reply the received proposal on the negotiating option δ. Then, the negotiation process accepts the proposal only when the next expected state provided by δ has a better evaluation than the current aspiration of the agent:

Ensure: $f : S \mapsto [0,1]$ is the negotiation state evaluation function
Ensure: s is the current negotiation state
  $s' \leftarrow next(s,\delta)$ {computes the next expected state}
  if $f(s') > \mathcal{A}(s,t)$ **then**
    accept(δ) {the interface module sends a message accepting δ}
  else
    reject(δ) {the interface module sends a message rejecting δ}
  end if

Finally, when the timeout is reached, the highest basic intention is to *executeActions*. Then, the negotiation protocol automatically cancel all ongoing negotiations and the HANA negotiation process selects the best complete plan from the plan ranking and executes it:

$\bar{P}_\alpha^t \leftarrow \{p | p \in \bar{P}_\alpha^\omega \text{ and } P_\alpha^t(p) \neq \bot\}$
$\bar{p} \leftarrow \arg\max_{\bar{p} \in \bar{P}_\alpha^t} E[\mathcal{U}_\alpha(\omega, \sigma(\mathcal{B}_\alpha^t, \{\langle \alpha, a, 1\rangle | a \in \bar{p}\})$ {available complete plan with highest utility}
execute($\bar{p}$) {the interface module executes all actions in $\bar{p}$}

Other intentions can be used by the negotiation strategy when desired. For example, a HANA agent with a trust model incorporated can generate the intention to *not negotiate with agent* β. In that case, Algorithm 6 should be modified to reject any proposal from β. And the plan evaluation of the plan search module should be modified to poorly evaluate joint plans with plans including β. The HANA architecture is designed to be able to incorporate new models in the world module and adjust the rest of the components. The architecture is flexible and allows to create a high diversity of different agents.

7. Discussion and future work

In this paper we have proposed HANA, an architecture for agents involved in Resource Negotiation Problems (RNP). These problems consider realistic multiagent scenarios where agents need to bilaterally negotiate joint plans of action possibly with humans repeatedly throughout time. The aim

of introducing RNP is to challenge the research community to study complex real negotiation scenarios involving humans. Specially those where the negotiation is about actions. The majority of current work focusses on the negotiation of a fixed number of issues. Codifying an RNP as a multi-issue negotiation problem (every resource being an issue and the values being the concrete action on the resource) would force the negotiation process to deal with very large and sparse negotiation objects. This would not be a practical approach. In [37] a classification of automated negotiation is made that does not consider plans as possible negotiation objects. Although RNPs are difficult problems we think that the automated negotiation research topic is mature enough to start facing the challenge.

One of the most compelling issues when researching in realistic scenarios with humans is experimentation. To support this task we have built the DipGame testbed [15] based on The Diplomacy Game[16] facilitating the development of software agents and the execution of experiments. This initiative is gaining relevance in the Multiagent Systems community. Several research departments from around the world are currently using DipGame and the number of registered users is increasing —50 user registrations during last two months.[17]

The architecture that we propose in this paper includes a BDI agent model. This BDI model deals with uncertainty using graded beliefs, desires and intentions [19]. The architecture is designed with the negotiation and the plan search modules executing concurrently. The negotiation module uses the best plans found so far by the search module to generate negotiation options, and the plan search module uses the commitments derived from proposals accepted by the negotiation module to prune the search space and improve the evaluation of the plans. HANA is a modular architecture that can be easily extended incorporating new behaviour models of personality, trust, relationships, etc. The idea of this architecture is to provide a skeleton for agents. Then, by fleshing out this skeleton, researchers in negotiation can easily integrate different behaviour models to build an agent with good skills in order to negotiate with humans.

In the LOGIC negotiation model agents act in response to a *need* [9]. Whenever there is a need, a LOGIC agent selects whom to negotiate with,

[16]The infrastructure of the DipGame testbed is available at `http://www.dipgame.org`.

[17]Only confirmed users are taken into account. Statistics at the end of January 2012.

prepares a negotiation strategy with relevant information (legitimacy) and computes the acceptable options. Differently from our work, no information is provided about how the negotiation options are generated from the needs. Here, we also provide details about how to evaluate options. LOGIC, as most negotiation methods, assumes that all possible negotiation partners have similar capabilities and thus selecting a partner does not depend of the concrete proposal to make. Here, instead, proposals determine which agents we should interact with as the capabilities of the negotiation partners can be different.

Our work is similar to [38] in that HANA also studies negotiation of joint plans of action involving several agents, is based on BDI agent models and assumes uncertainty in the actual outcome of plans. However, differently from [38], HANA agents can deal with uncertainty in their mental state by the use of a graded BDI model, it assumes the collaborations to be sporadical and it gives details of how plans are formulated and options negotiated. Other interesting works that study negotiation and planning in multiagent systems are [39, 40]. Both provide a logical framework and make assumptions (e.g. turn-taking dialogues and shared goals) that are difficult to find in human-aware scenarios. HANA focus on those scenarios and provide a flexible architecture for automated negotiation agents.

In the future we would like to study the use of a personality model like the one presented in [41] to select how the agents should emotionally react to observations by generating appropriate intentions. Analysing the personality of other agents is specially challenging as our agent counterparts can be humans. Another interesting model to incorporate is the relationship model described in [9] where intimacy levels are computed for each other agent and some strategies to reach a certain desired intimacy level are provided. The next behaviour model we plan to extend the architecture with is a trust model. The concept of trust is really important to perform negotiations. In environments where there is no trust among agents, negotiation cannot take place as the negotiators will not be able to belief on the actual execution of the agreement. We plan to incorporate a version of Sierra and Debenhams' trust model [42] (i) to provide beliefs on the agent's trust on other agents, (ii) to update the belief set from reached commitments, (iii) to provide the intentions to negotiate or not with a given agent, and (iv) to exclude from the plan ranking those joint plans with actions assigned to agents whom the HANA agent distrusts.

Acknowledgments

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. This work is also supported under project CBIT, which is funded by Spain's Ministry of Science and Innovation under grant number TIN2010-16306 and under ERA-NET project ACE.

References

[1] S. S. Fatima, M. Wooldridge, N. R. Jennings, An agenda-based framework for multi-issue negotiation, Artificial Intelligence 152 (1) (2004) 1 – 45.

[2] R. Lin, S. Kraus, J. Wilkenfeld, J. Barry, Negotiating with bounded rational agents in environments with incomplete information using an automated agent, Artificial Intelligence 172 (2008) 823–851.

[3] N. Matos, C. Sierra, N. R. Jennings, Negotiation strategies: An evolutionary approach, in: Proc. Int. Conf. on Multi-agent Systems (ICMA'S 98), 1998, pp. 182–189.

[4] K. V. Hindriks, C. Jonker, D. Tykhonov, Towards an open negotiation architecture for heterogeneous agents, in: Proc. of the 12th Int. Workshop on Cooperative Information Agents, Prague, 2008, pp. 264–279.

[5] J. Rosenschein, G. Zlotkin, Rules of Encounter, MIT Press, 1998.

[6] R. Lin, S. Kraus, From research to practice: Automated negotiations with people, http://u.cs.biu.ac.il/ linraz/Papers/linetal-practice.pdf.

[7] V. L. Smith, Constructivist and ecological rationality in economics, The American Economic Review 93 (3) (2003) 465–508.

[8] J. Debenham, C. Sierra, An agent supports constructivist and ecological rationality, in: Proc. 2009 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology, Milano, 2009, pp. 255–258.

[9] C. Sierra, J. Debenham, The logic negotiation model, in: Proc. of 6th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2007), 2007, pp. 1026–1033.

[10] D. Fessler, K. J. Haley, Genetic and cultural evolution of cooperation, 2003, Ch. The strategy of affect: emotions in human cooperation, pp. 7–36.

[11] Y. Gal, B. Grosz, S. Kraus, A. Pfeffer, S. Shieber, Agent decision-making in open mixed networks, Artificial Intelligence 174 (2010) 1460–1480.

[12] F. A. Hayek, The Fatal Conceit : The Errors of Socialism (The Collected Works of F. A. Hayek), University Of Chicago Press, 1991.

[13] A. Jøsang, R. Ismail, C. Boyd, A survey of trust and reputation systems for online service provision, Decision Support Systems 43 (2) (2007) 618–644.

[14] D. Kim, D. Ferrin, H. Rao, A trust-based consumer decision-making model in electronic commerce: The role of trust, perceived risk, and their antecedents, Decision Support Systems 44 (2) (2008) 544–564.

[15] A. Fabregues, C. Sierra, Dipgame: a challenging negotiation testbed, Journal of Engineering Applications of Artificial Intelligence 24 (2011) 1137–1146.

[16] A. Fabregues, D. Navarro, A. Serrano, C. Sierra, Dipgame: a testbed for multiagent systems (demonstration), in: Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010), 2010, pp. 1619–1620.

[17] R. Sharp, The Game of Diplomacy, 1978, http://www.diplom.org/ diparch/god.htm.

[18] M. Minsky, The emotion machine: from pain to suffering, in: Proc. of the 3rd Conf. on Creativity & cognition (C&C '99), ACM, 1999, pp. 7–13.

[19] A. Casali, L. Godo, C. Sierra, A graded bdi agent model to represent and reason about preferences, Artificial Intelligence 175 (2011) 1468–1478.

[20] R. H. Bordini, J. F. Hübner, M. Wooldridge, Index, 2007, pp. 269–273.

[21] M. Winikoff, Jack intelligent agents: An industrial strength platform, in: Multi-Agent Programming, Vol. 15, Springer, 2005, pp. 175–193.

[22] A. S. Rao, Agentspeak(l): Bdi agents speak out in a logical computable language, in: Proc. of the 7th European workshop on Modelling autonomous agents in a multi-agent world (MAAMAW '96), 1996, pp. 42–55.

[23] M. d'Inverno, M. Luck, Engineering agentspeak(l): A formal computational model, Logic and Computation 8 (3) (1998) 233–260.

[24] K. Hindriks, M. d'Inverno, M. Luck, Architecture for agent programming languages, in: Proc. of the 14th European Conf. on Artificial Intelligence (ECAI 2000), 2000, pp. 363–367.

[25] M. Dastani, 2apl: a practical agent programming language, Autonomous Agents and Multi-Agent Systems 16 (3) (2008) 214–248.

[26] A. S. Rao, M. P. Georgeff, Modeling Rational Agents within a BDI-Architecture, in: Proc. of the 2nd Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'91), 1991, pp. 473–484.

[27] S. Parsons, P. Giorgini, An approach to using degrees of belief in BDI agents, in: Proc. of the Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, 1999.

[28] P. Pilotti, A. Casali, C. Chesñevar, An approach to automated agent negotiation using belief revision, in: Proc. of the12th Argentine Symposium on Artificial Intelligence (40th JAIIO), Córdoba, Argentina, 2011, pp. 202 – 212.

[29] M. E. Bratman, Intention, Plans, and Practical Reason, Cambridge University Press, 1999.

[30] A. S. Rao, M. P. Georgeff, Decision procedures for BDI logics, Journal of Logic and Computation 8 (3) (1998) 293–342.

[31] C. E. Alchourròn, P. Gärdenfors, D. Makinson, On the logic of theory change: Partial meet contraction and revision functions, Journal of Symbolic Logic 50 (1985) 510–530.

[32] S. O. Hansson, A Textbook of Belief Dynamics: Solutions to Exercises, Kluwer Academic Publishers, Norwell, MA, USA, 2001.

[33] A. Fabregues, J. Madrenas, C. Sierra, J. Debenham, Supplier performance in a digital ecosystem, in: Proc. of the IEEE Int. Conf. on Digital Ecosystems and Technologies (IEEE-DEST 2009), Istanbul, 2009, pp. 466–471.

[34] W. Revelle, K. R. Scherer, Personality and emotion, in: Oxford Companion to the Affective Sciences, Oxford University Press, 2010.

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

[36] L. P. Kaelbling, M. L. Littman, A. R. Cassandra, Planning and acting in partially observable stochastic domains, Artif. Intell. 101 (1998) 99–134.

[37] R. Buttner, A classification structure for automated negotiations, in: Proc. of the 2006 IEEE/WIC/ACM Int. Conf. on Web Intelligence and Intelligent Agent Technology (WI-IATW '06), IEEE Computer Society, Washington, DC, USA, 2006, pp. 523–530.

[38] B. J. Grosz, L. Hunsberger, S. Kraus, Planning and Acting Together.

[39] P. Pardo, P. Dellunde, L. Godo, Argumentation-based negotiation in t-delp-pop, in: Proc. of the 14th Int. Conf. of the Catalan Association for Artificial Intelligence (CCIA 2011), Vol. 232, 2011, pp. 177–186.

[40] A. C. Kakas, P. Torroni, N. Demetriou, Agent planning, negotiation and control of operation, in: of the 16th European Conf. on Artificial Intelligence (ECAI 2004), 2004, pp. 28–32.

[41] L. Peña, J.-M. Peña, S. Ossowski, Representing emotion and mood states for virtual agents, in: Proc. of the 9th German Conf. on Multiagent system technologies, 2011, pp. 181–188.

[42] C. Sierra, J. K. Debenham, Trust and honour in information-based agency, in: Proc. of the 5th Int. Conf. on Autonomous Agents and Multi-agent Systems (AAMAS 2006), 2006, pp. 1225–1232.