Decision Support Systems 60 (2014) 18–28

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

HANA: A Human-Aware Negotiation Architecture

Angela Fabregues *, Carles Sierra

Artificial Intelligence Research Institute (IIIA-CSIC), Campus Universitat Autònoma de Barcelona, E-08193 Bellaterra, Spain

ARTICLE INFO

Available online 5 June 2013

Keywords:
Multiagent systems
Agent architecture
Automated negotiation
Practical reasoning
Search&negotiation
Diplomacy Game

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.

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 [13,26]. In most previous work autonomous software agents, usually selfish, interact using utility maximisation strategies [20,27,37]. 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 [25]. This is in part because humans do not follow a *constructivist* sort of rationality [8,41]. For instance, human decisions depend a lot on their social relationships [40], on emotions [14] and are contextualised in a particular culture [15].

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 [18] as developed in the LOGIC theory of agency [40] 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 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 is 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

* Corresponding author.
E-mail addresses: fabregues@iiia.csic.es (A. Fabregues), sierra@iiia.csic.es (C. Sierra).

0167-9236/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.dss.2013.05.017

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 [21,24]. 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 A that control resources R and are always in one of several possible states.

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

$\langle \alpha, r \rangle \in \omega$ 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 $A = \mathrm{A} \times Op \times R$.*

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* comp(a, b), *if and only if $op_a = op_b$ implies $r_a \neq r_b$.*

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* feasible(a, ω), *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* feasible(p, ω), *if and only if for all $a, b \in p$,* feasible(a, ω) *and* 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* comp(p, q), *if and only if their union is feasible, that is,* comp$(p, q) \iff p \cup q \in P^{\omega}$.

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* feasible(ω, 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 $\overline{P}^{\omega} \subseteq P^{\omega}$ and by $\overline{P}^{\omega}_{\alpha}$ 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 labelling the arcs between states.

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

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

where:

- *A is a set of agents*
- *R is a set of resources*
- *Op is a set of operators*
- *$W = 2^{\mathrm{A} \times R}$ is a set of states*
- *$P = 2^{\mathrm{A} \times Op \times R}$ 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 $p \in \overline{P}^{\omega}$*
- *$\omega_0 \in W$ is the initial state*
- *$W_f \subseteq W$* is the set of final states.

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