Specifying Agent Interaction Protocols

Sonia V. Rueda María Vanina Martinez [1] Guillermo R. Simari

Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA) [2]
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur. Av. Alem 1253, (8000) Bahía Blanca, Argentina
Tel: ++54 291 4595135 - Fax: ++54 291 4595136
{svr,mvm,grs}@cs.uns.edu.ar

Abstract

Argumentation-based negotiation is an adequate alternative for modelling situations in which agents have limited information and bounded capacities. In this research line, each agent elaborates arguments as part of its own planning process and also to justify its proposals, counter-proposals and rejections during the negotiation process.

Our proposal analyzes and compares three tools for the specification of interaction protocols: finite state machines, UML, and dialogues games.

1 Introduction and Background

Negotiation is a fundamental activity in a multi-agent system. The members of the system negotiate in order to coordinate their activities and to distribute resources and tasks trying to reach a state acceptable to all of them.

The negotiation models vary depending on the system's characteristics. If all the members are part of an organization, the relationship among them can be a collaborative one, even when it will frequently be necessary for them to interact in order to align their interests. The group can also be composed by homogeneous or heterogeneous agents. In the former case, all the members share the same view of the world and they have identical capacities. In a heterogeneous group, agents will in general have distinct views of the world and different abilities.

In our work we adopt the BDI model for representing the mental attitudes of each member of the group. The individual knowledge of each agent is conformed by its specific knowledge and the knowledge shared with other members in the group. Each agent will reason using the facts available to it. As it is proposed in [6], the shared knowledge is distributed among *pairs* of agents; therefore, even if each agent's view of the world is consistent, different members of the system can have different views. The group is heterogeneous, and each agent's goals are tied to their abilities. Despite their differences, all of the members in the organization are

[1]Partially supported by *Secretaría de Ciencia y Tecnologa, Universidad Nacional del Sur*, CONICET, and by the *Agencia Nacional de Promoción Científica y Tecnológica* (PICT 202 Nro 13096 and PICT 2003 15043)

[2]Member of the IICyTI (Instituto de Investigación Científica en Ciencia y Tecnología Informática).

autonomous and rational entities with a collaborative attitude. When their beliefs and abilities do not suffice to reach their goals they request collaboration, starting a negotiation process.

Argument-based negotiation is a suitable alternative for modelling situations in which agents have limited information and bounded capacities [5]. During the process, the participants acquire information, but it is also possible for them to reach a point in which they must revise their plans and even modify their preferences in order to be able to reach an agreement. In our proposal each agent elaborates arguments as part of their own planning processes [1] and to justify their proposals, counter-proposals, and rejections during the negotiation process. Interaction is implemented through dialogues among pairs of agents, and the set of dialogues generated inside the same negotiation process conforms a *conversation*.

The literature offers different alternatives for specifying interaction protocols in multi-agent systems. The research line proposed in this work is oriented towards the analysis and comparison of three alternatives: finite state machines, UML, and dialogue games.

2 Interaction Protocols

A multi-agent system consists of an organized group of agents which interact with each other. This interaction is generally regarded as the foundation for cooperative and competitive behaviour in autonomous agents. The term *interaction protocols* is used in reference to sets of rules that guide interactions. Negotiation establishes a form of interaction among agents trying to reach a mutually acceptable agreement.

Negotiation can be thought of as a distributed search process over the space of potential agreements. The process is not linear, and therefore the space is not reduced until the solution is reached because but it can move and even incorporate new points. In most cases, each agent knows only part of the search space and, within it, there is only a portion which satisfies its expectations. Each agent has a specific set of points within the space of agreements that are acceptable for it. The search is successful when an agreement space is reached, that is, there exists a nonempty intersection among the individual spaces. The process ends when the search ends, regardless of its success or failure.

In a simple interaction protocol the agents elaborate, accept, or reject proposals. This approach is not adequate when negotiation is viewed as a search process. In this case, the receiver of a proposal must be able not only to accept or reject the proposal, but also to guide the process through its answer. Agents perform proposals and counter-proposals elaborating *arguments* which intend to persuade other agents [2]. The interaction language must offer a set of primitives suitable for expressing proposals and counter-proposals, offering arguments and expressing the interest level that each agent assigns to each collaboration request. The research line presented in this article is complemented with the definition of an interaction language which takes these aspects into consideration.

In our work, when the negotiation ends successfully, the shared knowledge is modified with the incorporation of new beliefs. Since the shared knowledge is distributed among pairs of

agents, the modification initially affects only two agents. However, the negotiation process may have involved various members of the group.

The proposal is, then, for the negotiation to generate a *conversation* in which more than two agents will probably take part. Nevertheless, interaction is always performed through *dialogues* which only link two agents. Modifications made by both agents to their shared knowledge are only performed after a global agreement has been reached. Each agent must maintain the commitment implied in each dialogue until it has been freed from it, or an acceptable agreement has been reached. The interaction protocol specification must include, at least, the following elements:

- Types of participants.
- Interaction states.
- Events which trigger states changes.
- Valid actions given the participant and the state.

Protocol specification languages can be analyzed and compared according to the facilities they offer for expressing these elements. There has been a variety of protocol specification languages proposed differing in the level of abstraction for which they were designed. None of them, however, was able to gain general acceptance. This is due, in part, to the fact that these languages only provide text-based representations for the protocols. These representation are not adequate, mainly in complex protocols, because the flow of control in the protocol is obscured. Graphic languages therefore arise as an alternative to text-based representations. Graphic languages have the advantage of providing a set of symbols and mechanisms with well-defined semantics so that software designers can express and exchange ideas in a friendlier manner.

3 Finite State Machines

Interaction protocols are often represented by finite state machines, which can be defined by a graph including a finite set Q of states, an initial state q_0, a finite set A of possible locutions, and a transition function $T : Q \times A$. The states represent points in which decisions are made; the next transition is chosen among the states that are associated with the current one, and each of these transitions represents a step of the interaction role. The reception of a message or the expiration of a timeout correspond to an event. The content of the received message is checked by a condition, as is the value of an interaction role attribute.

This specification form is adequate when the communication language offers a reduced set of primitives. When the number of locutions is large, the amount of possible interactions grows significantly and it is very difficult to reflect all the possible combinations in a finite state machine.

4 UML

UML is currently one of the most powerful graphic design languages for describing software systems. A unified language increases the interoperability among software design tools making them independent from the environment in which they were developed, and they can be assembled in a context different from the one in which they were conceived.

UML's activity diagrams can be used in specifying the interaction among the agents in a system. Computations are expressed in terms of states and the progression through them. Action states are atomic entities similar to atomic statements in a programming language. In contrast, activity states represent a *collection* of atomic states that can be decomposed into atomic ones; the execution of an activity can be interrupted between any two subsequent states.

Transitions in an activity diagram provide the links between states and indicate the flow of control in the diagram. Guard conditions can be used in transitions to affect the flow of control; the transition can only be fired if this condition is true. Special states are introduced in order to represent the beginning and the end of an activity diagram. Branching elements representing decision points are provided so that the flow of control can become non-linear. In UML, threads can be modeled using two structural elements: the *fork* operation and the *join* element. The fork operation splits a single thread of execution into two or more threads that are subsequently executed in parallel. A join barrier can be used to synchronize parallel threads of execution, waiting until all incoming threads have arrived before proceeding with the single master thread. Because of the fact that activity diagrams tend to become somewhat confusing as they grow in size, UML activity diagrams can contain *swimlanes* that are used to partition an activity diagram into several conceptually related parts.

A large body of research proposes an extension of UML, increasing its expressive power in order to support concepts which are specifically oriented towards interaction among agents. On the other hand, other authors consider that it is important to avoid the proliferation of UML dialects developed for different application domains, and maintain only one general graphic language.

5 Dialogue Games

Dialogue games have existed for centuries to express argumentation. Today, this formalism can be used to specify meaningful interaction between dialogical partners by following the rules of an individual dialogue. The interaction between two or more *players* is defined by means of a formal dialogue game, in which locutions are considered moves. The rules specify which locutions are permitted under what circumstances, and which responses are possible. There are different types of dialogue game rules, as proposed in [3]:

- Commencement and termination: define the circumstances under which the dialogue begins and ends.

- Locutions: specify the nature of the utterances permitted in the dialogue.
- Combination: define the dialogical contexts under which a particular locution is allowed.
- Commitment: define the circumstances under which a participant expresses dialogical commitment to a proposition.

This formalism provides a unifying framework that represents different types of dialogues, each of which has a simple semantics. In an interaction protocol based on dialogue games, it is possible to identify appropriate speech acts and to define constraints on their utterances.

6 Conclusions

There is currently no general agreement on a unique model of negotiation that can be used in any application domain. The argumentation-based approach has become popular for environments in which agents have internal and changing motivations, as well as different decision mechanisms. The exchange of arguments can improve the negotiation process, allowing a faster convergence among interests that are initially different. However, negotiation protocols based on argumentation are complex, and their specification demands the use of powerful tools.

Our work is oriented towards the analysis and comparison of three of the specification forms proposed by the current literature as tools for expressing a protocol based on argumentation among BDI agents. The analysis is oriented towards each tool's capacity for expressing different types of dialogues which can emerge, and the way in which they can be structured.

References

[1] García, A. J., and Simari, G. R. Defeasible logic programming: An argumentative approach. In *Theory and Practice of Logic Programming* (2004), pp. 95–138.

[2] Jennings, N. R., Parsons, S., Noriega, P., and Sierra, C. On argumentation-based negotiation. In *Proc. of the Int. Workshop on Multi-Agent Systems* (Boston, USA, 1998).

[3] Mcburney, P., Eijk, R. M. V., Parsons, S., and Amgoud, L. A dialogue game protocol for agent purchase negotiations. *Autonomous Agents and Multi-Agent Systems 7*, 3 (2003), 235–273.

[4] Odell, J., Parunak, H. V. D., and Bauer, B. Representing agent interaction protocols in uml. In *First international workshop, AOSE 2000 on Agent-oriented software engineering* (Secaucus, NJ, USA, 2001), Springer-Verlag New York, Inc., pp. 121–140.

[5] Rahwan, I. Argumentation-based negotiation, 1985.

[6] Rueda, S. V., García, A. J., and Simari, G. R. Alternativas para la representación del conocimiento compartido entre agentes colaborativos. In *Proc. of the VIII Congreso Argentino de Ciencias de la Computación* (2004), pp. 1508–1519.