Reasoning about self and others: communicating agents in a modal action logic

Matteo Baldoni, Cristina Baroglio, Alberto Martelli, and Viviana Patti

Dipartimento di Informatica — Università degli Studi di Torino
C.so Svizzera 185, I-10149 Torino (Italy)
E-mail: {baldoni,baroglio,mrt,patti}@di.unito.it

Abstract. We propose an approach to reasoning about conversation protocols within the framework of a logic-based *agent language*. We show how to embed a theory of communicative actions in the framework of a modal logic of action and beliefs, to specify software agents that, situated in a multi-agent environment, can interact with one another by a speech act based communication mechanism. Agents have their own local beliefs on the world and on the other agents mental state. Complex communicative behaviors can be specified as conversation protocols, and agents can reason on the belief dynamics caused by communications, before committing to a given interaction.

1 Introduction and motivations

In the last few years, great attention has been devoted to the issue of communication and dialogue among agents, in the context of a formal approach to the theory of agency [13]. In particular, the diffusion of open multi-agent systems has led the agent community to focus on the creation of *standardized communication languages* (ACL), that, having an explicit, general and well-defined semantic, could be used by heterogeneous agent programs and give an answer to the interoperability issue [15, 17]. In this framework, while a lot of work has been done in defining the semantics of the agent speech acts, those semantics aspects of communication that are related to the conversational context, in which a speech act occurs, started being investigated only recently [20, 22]. Moreover, although formal models of speech acts take into account the mental state of other agents, the approaches to communication taken in the practical setting of agent languages [24, 12] do not account for this aspect and do not permit to model individual agents that *subjectively* reason about effects of communication on the mental state of their interlocutors.

The capability of reasoning about conversations is useful in many application areas. Let us consider, as an example, a user personal assistant, i.e. a software agent that searches the web to find *web services*, according to a user's specification. The currently available languages for describing web services (e.g. DAML-S [9], WSDL [8]) base descriptions on the lists of inputs/outputs required/returned by the service. The agent matches the user's request ("book two tickets at a cinema where they show Akira but do not give my credit card number") to the

descriptions of the available services and selects one that satisfies all the conditions. Some of these conditions ("do not give my credit card number"), actually concern the way in which the *interaction* between the service provider and the personal assistant should be carried on. Standard description languages do not allow the representation of behavioral information [7] but if a formal specification of the interaction protocol was available, the personal assistant could *reason about* the change caused by a conversation to its own belief state and make rational assumptions on the change caused to the provider beliefs. In the application framework, the agent could, then, either verify if the interaction may be *personalized* by following an execution path that satisfies all the user's requirements, or, when this is not possible, decide to search for another provider.

In this work, we face the problem of describing and reasoning about conversation protocols in an *agent logic programming* setting, by extending the logical framework of the agent language DyLOG [5] so to deal also with *communicative behaviors*. DyLOG is a logic programming language for specifying and reasoning about the behavior of rational agents, based on a modal logic for reasoning about *actions* and *beliefs*, that has successfully been used in the development of adaptive web applications [4]. It permits to define complex actions and sensing actions. Agents programmed in DyLOG choose a course of actions, conditioned by their beliefs about the world, and use sensors for acquiring knowledge. We present an extension of the language, in which a *communication kit*, including both primitive speech acts and conversation protocols, has been integrated. Such an extension is based on an agent theory, in which agents have *local* beliefs about the world and about the mental state of the other agents, and where communications are modelled as actions that transform the interlocutor mental state. Our account of communication aims at coping with two main aspects: the state change caused by a communicative act on an agent local beliefs, and the decision strategy used by the agent for sending suitable answers to a received communication. To these aims, the semantics of primitive speech acts is described in terms of effects on the mental state both in case the agent is the sender and in case it is the recipient. Moreover, in the line of [20], we use conversation protocols as decision procedures that allow agents to suitably respond to communications. Conversation protocols are built upon speech acts, and specify communication patterns for agent conversations. We took a subjective representation of such protocols, by making hypothetical assumptions on the *other*'s answers. They have been easily integrated with the other policies that specify the agent behavior, being both represented as complex actions by DyLOG procedure axioms. We provide a goal-directed proof procedure in order to support agent's reasoning and planning in presence of communication. This procedure allows an agent to reason about the interaction that it is going to enact with another agent, with the aim of proving if there is a possible execution of the *communication protocol*, after which a set of beliefs of interest (goal) will be true in its mental state.

The article is orgnized as follows: Section 2 introduces DyLOG with a particular attention to the tools that it offers for dealing with communication; Section 3 briefly shows the solution to the persistency problem that we adopted; in

Section 4 we describe the techniques applied for reasoning in presence of communication; an example application follows, and the article is concluded by a contextualization of the work in the literature and a few considerations.

2 The agent language

The agent language accounts both for atomic and complex actions, or procedures. Atomic actions are either world actions, affecting the world, or mental actions, i.e. sensing or communicative actions which only affect the agent beliefs. The set of atomic actions consists of the set $\mathcal{A}$ of the world actions, the set $\mathcal{C}$ of communicative acts, and the set $\mathcal{S}$ of sensing actions. For each atomic action a and agent ag_i we introduce the modalities $[a^{ag_i}]$ and $\langle a^{ag_i}\rangle$. $[a^{ag_i}]\alpha$ means that α holds after every execution of action a by agent ag_i; $\langle a^{ag_i}\rangle\alpha$ means that there is a possible execution of a (by ag_i) after which α holds.For each atomic action a in $\mathcal{A}\cup\mathcal{C}$ and agent ag_i we also introduce a modality $Done(a^{ag_i})$ for expressing that a has been executed. $Done(a^{ag_i})\alpha$ is read "a has been executed by ag_i; before its execution, α was true"[1]. The modality $\Box$ denotes formulas that hold in all the possible agent mental states. Our formalization of complex actions draws considerably from dynamic logic for the definition of action operators like sequence, test and non-deterministic choice. However, differently than [19], we refer to a *Prolog-like* paradigm: procedures are defined by means of (possibly recursive) Prolog-like clauses. For each procedure p, the language contains also the universal and existential modalities $[p]$ and $\langle p\rangle$. The mental state of an agent is described in terms of a consistent set of *belief formulas*. We enriched the belief state of a DyLOG agent by allowing also nested beliefs, for representing what other agents believe and reasoning on how they can be affected by communicative actions. We use the modal operator $\mathcal{B}^{ag_i}$ to model the beliefs of agent ag_i. The modality $\mathcal{M}^{ag_i}$ is defined as the dual of $\mathcal{B}^{ag_i}$ ($\mathcal{M}^{ag_i}\varphi \equiv \neg\mathcal{B}^{ag_i}\neg\varphi$). Intuitively $\mathcal{M}^{ag_i}\varphi$ means that ag_i consider φ possible.

All the modalities of the language are normal; $\Box$ is reflexive and transitive, its interaction with action modalities is ruled by $\Box\varphi \supset [a^{ag_i}]\varphi$. The epistemic modality $\mathcal{B}^{ag_i}$ is serial, transitive and euclidean. The interaction of the $Done(a^{ag_i})$ modality with other modalities is ruled by: $\varphi \supset [a^{ag_i}]Done(a^{ag_i})\varphi$ and $Done(a^{ag_j})\varphi \supset \mathcal{B}^{ag_i}Done(a^{ag_j})\varphi$ (awareness), with $ag_i = ag_j$ when $a^{ag_i} \notin \mathcal{C}$.

2.1 The agent theory

In the line of [5] the *behavior* of an agent ag_i can be specified by a domain description, which includes, besides a specification of the agent *belief state*: (1) action and precondition laws for describing the *atomic world actions* in terms of their preconditions and effects on the executor's mental state; (2) sensing axioms for describing *atomic sensing actions*; (3) procedure axioms for describing *complex behaviors*.

[1] $Done(a^{ag_i})\top$ is read "the action a has been executed by agent ag_i".

Belief state Agents are individuals, each having a mental state: its *subjective* point of view on a dynamic domain. Then, we do not model the real world but only the internal dynamics of each agent in relation to the changes caused by actions. A mental state is a set of belief formulas (*belief state*), intuitively it contains what ag_i (dis)believes about the world and about the other agents. A belief state is a complete and consistent set of rank 1 and 2 belief fluents, where a *belief fluent* F is a belief formula $\mathcal{B}^{ag_i}L$ or its negation. L denotes a *belief argument*, i.e. a *fluent literal* (f or $\neg f$), a *done fluent* ($Done(a^{ag_i})\top$ or its negation), or a belief fluent of rank 1 ($\mathcal{B}l$ or $\neg\mathcal{B}l$). We use l for denoting attitude-free fluents: a fluent literal or a done fluent. Consistency is guaranteed by the seriality of the $\mathcal{B}^{ag_i}$ modalities[2]. In essence a belief state provides, for each agent ag_i, a three-valued interpretation of all the possible belief arguments L: each L is either *true*, *false*, or *undefined* when both $\neg\mathcal{B}^{ag_i}L$ and $\neg\mathcal{B}^{ag_i}\neg L$ hold. In the following we use $\mathcal{U}^{ag_i}L$ for expressing the ignorance of ag_i about L.
World actions are described by their preconditions and effects on the *actor*'s mental state; they trigger a revision process on the actor's beliefs. Formally, *action laws* describe the conditional effects on ag_i's belief state of an atomic action $a \in \mathcal{A}$, executed by ag_i itself. They have the form:

$$\Box(\mathcal{B}^{ag_i}L_1 \wedge \ldots \wedge \mathcal{B}^{ag_i}L_n \supset [a^{ag_i}]\mathcal{B}^{ag_i}L_0) \quad (1)$$

$$\Box(\mathcal{M}^{ag_i}L_1 \wedge \ldots \wedge \mathcal{M}^{ag_i}L_n \supset [a^{ag_i}]\mathcal{M}^{ag_i}L_0) \quad (2)$$

Law (1) states that if ag_i believes the preconditions to an action a in a certain epistemic state, after a execution, ag_i will also believe the action's effects. (2) states that when the preconditions of a are unknown to ag_i, after the execution of a, ag_i will consider unknown also its effects[3]. *Precondition laws* specify mental conditions that make an action in $\mathcal{A} \cup \mathcal{C}$ executable in a state. They have form:

$$\Box(\mathcal{B}^{ag_i}L_1 \wedge \ldots \wedge \mathcal{B}^{ag_i}L_n \supset \langle a^{ag_i}\rangle\top) \quad (3)$$

ag_i can execute a when the precondition fluents of a are in ag_i's belief state.
Sensing Actions produce knowledge about fluents; they are defined as non-deterministic actions, with unpredictable outcome, formally modelled by a set of *sensing axioms*. If we associate to each sensing action s a set $dom(s)$ of literals (domain), when ag_i executes s, it will know which of such literals is true:

$$[s]\varphi \equiv [\bigcup_{l \in dom(s)} s^{\mathcal{B}^{ag_i}l}]\varphi \quad (4)$$

$\cup$ is the choice operator of dynamic logic and $s^{\mathcal{B}^{ag_i}l}$, for each $l \in dom(s)$, is an *ad hoc* primitive action, that probes one of the possible outcomes of the sensing.

[2] A belief state is *not consistent* when it contains: a belief $\mathcal{B}^{ag_i}l$ and its negation, or the belief formulas $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i}l$ and $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i}\neg l$, or the belief formulas $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i}l$ and $\mathcal{B}^{ag_j}\neg\mathcal{B}^{ag_i}l$.

[3] Laws of form (2) allow actions with non-deterministic effects, that may cause a *loss* of knowledge, to be specified.

Complex actions We specify agent complex behaviors by means of *procedure definitions*, built upon other actions. Formally, a complex action is defined by means of a collection of *inclusion axiom schema* of our modal logic, of form:

$$\langle p_0 \rangle \varphi \subset \langle p_1; p_2; \ldots; p_m \rangle \varphi \tag{5}$$

p_0 is a procedure name and the p_i's ($i = 1, \ldots, m$) are either procedure names, atomic actions, or test actions; the operator ";" is the sequencing operator of dynamic logic. Procedure definitions may be recursive and procedure clauses can be executed in a goal directed way, similarly to standard logic programs.

2.2 Communication

The integration of a *communication theory* in the general agent theory is obtained by adding further axioms and laws to ag_i's domain description. In this section we will introduce a communication kit that allows the specification of communicative behaviors.

Speech Acts Communication primitives are atomic actions, described in terms of preconditions and effects on the agent mental state. They have the form $\mathsf{speech\_act}(sender, receiver, l)$, where *sender* and *receiver* are agents and l is either a fluent literal or a done fluent. Such actions can be seen as special mental actions, affecting both the sender's and the receiver's mental state. In our model we focused on the *internal representation*, that agents have of each speech act, by specifying ag_i's belief changes both when it is the sender and when it is the receiver. They are modelled by generalizing the action and precondition laws of form (1), (2), and (3), so to allow the representation of the effects of communications performed by other agents on ag_i mental state. Such a representation provides the capability of *reasoning about* conversation effects.

Speech act specification is, then, twofold: one definition holds when the agent is the sender, the other when it is the receiver. In the first case, the precondition laws contain some *sincerity condition* that must hold in the agent mental state. When ag_i is the receiver, the action is *always* executable. Let us consider some primitive speech acts from the standard agent communication language FIPA-ACL, and let us define them and their semantics within our framework:

$\mathsf{inform}(sender, receiver, l)$

a) $\Box(\mathcal{B}^{ag_i} l \wedge \mathcal{B}^{ag_i}\mathcal{U}^{ag_j} l \supset \langle \mathsf{inform}(ag_i, ag_j, l)\rangle \top)$

b) $\Box([\mathsf{inform}(ag_i, ag_j, l)]\mathcal{M}^{ag_i}\mathcal{B}^{ag_j} l)$

c) $\Box(\mathcal{B}^{ag_i}\mathcal{B}^{ag_j} authority(ag_i, l) \supset [\mathsf{inform}(ag_i, ag_j, l)]\mathcal{B}^{ag_i}\mathcal{B}^{ag_j} l)$

d) $\Box(\top \supset \langle \mathsf{inform}(ag_j, ag_i, l)\rangle \top)$

e) $\Box([\mathsf{inform}(ag_j, ag_i, l)]\mathcal{B}^{ag_i}\mathcal{B}^{ag_j} l)$

f) $\Box(\mathcal{B}^{ag_i} authority(ag_j, l) \supset [\mathsf{inform}(ag_j, ag_i, l)]\mathcal{B}^{ag_i} l)$

g) $\Box(\mathcal{M}^{ag_i} authority(ag_j, l) \supset [\mathsf{inform}(ag_j, ag_i, l)]\mathcal{M}^{ag_i} l)$

Clause (a) states that an inform act can be executed when the sender believes l and believes that the receiver does not know l. When ag_i is the sender it thinks

possible that the receiver will adopt its belief, although it cannot be certain - autonomy assumption (b)-. If it believes that ag_j considers it a trusted *authority* about l, it is confident that the receiver will adopt its belief (c). When ag_i is the receiver, it believes that l is believed by the sender ag_j (e), but it adopts l as an own belief only if it thinks ag_j is a trusted authority (f)-(g).

$\mathsf{queryIf}(sender, receiver, l)$
a) $\Box(\mathcal{U}^{ag_i} l \wedge \neg \mathcal{B}^{ag_i} \mathcal{U}^{ag_j} l \supset \langle \mathsf{queryIf}(ag_i, ag_j, l) \rangle \top)$
b) $\Box(\top \supset \langle \mathsf{queryIf}(ag_j, ag_i, l) \rangle \top)$
c) $\Box([\mathsf{queryIf}(ag_j, ag_i, l)] \mathcal{B}^{ag_i} \mathcal{U}^{ag_j} l)$

By queryIf ag_i asks ag_j if it believes that l is true. To perform a queryIf act, ag_i must ignore l and it must believe that the receiver does not ignore l (a). After a queryIf act, the receiver will believe that the sender ignores l.

$\mathsf{refuseInform}(sender, receiver, l)$
a) $\Box(\mathcal{U}^{ag_i} l \wedge \mathcal{B}^{ag_i} Done(\mathsf{queryIf}(ag_j, ag_i, l)) \top \supset \langle \mathsf{refuseInform}(ag_i, ag_j, l) \rangle \top)$
b) $\Box(\top \supset \langle \mathsf{refuseInform}(ag_j, ag_i, l) \rangle \top)$
c) $\Box([\mathsf{refuseInform}(ag_j, ag_i, l)] \mathcal{B}^{ag_i} \mathcal{U}^{ag_j} l)$

By refuseInform an agent refuses to give an information it was asked for. The refusal can be executed only if: the sender ignores l and it believes that the receiver previously queried it about l. After a refusal the receiver believes that the sender ignores l.

Get Message Actions are used for *receiving* messages from other agents. We model them as a special kind of sensing actions, because from the agent perspective they correspond to queries for an external input, whose outcome is unpredictable. The main difference w.r.t. normal sensing actions is that they are defined by means of speech acts performed by the interlocutor. Formally, we use get_message actions defined by an axiom schema of the form:

$$[\mathsf{get\_message}(ag_i, ag_j, l)]\varphi \equiv [\bigcup_{\mathsf{speech\_act} \in \mathcal{C}_{\mathsf{get\_message}}} \mathsf{speech\_act}(ag_j, ag_i, l)]\varphi \quad (6)$$

Intuitively, $\mathcal{C}_{\mathsf{get\_message}}$ is a finite set of speech acts, which are all the possible communications that ag_i expects from ag_j in the context of a given conversation. We do not associate to a get_message action a domain of mental fluents, but we calculate the information obtained by looking at the effects of the speech acts in $\mathcal{C}_{\mathsf{get\_message}}$ on ag_i's mental state.

Conversation protocols We suppose individual speech acts to take place in the context of predefined conversation protocols [20] that specify communication patterns. Each agent has a subjective perception of the communication with other agents, for this reason each protocol has as many procedural representations as the possible roles in the conversation. Let us consider, for instance the yes_no_query protocol reported in Fig. 1, a simplified version of the FIPA Query Interaction Protocol [16]. The protocol has two complementary views, one to be followed for making a query ($\mathsf{yes\_no\_query}_Q$) and one for responding ($\mathsf{yes\_no\_query}_I$). In the following get_answer and get_start definitions are instances of the get_message axiom.

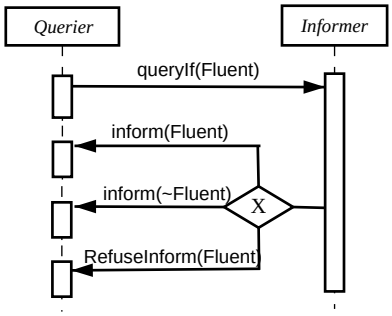

Fig. 1. The AUML graph [21] represents the communicative interactions occurring between the *querier* and the *informer* in the yes_no_query protocol.

$$\langle \mathsf{yes\_no\_query}_Q(Self, Other, Fluent)\rangle\varphi \subset \\ \langle \mathsf{queryIf}(Self, Other, Fluent); \mathsf{get\_answer}(Self, Other, Fluent)\rangle\varphi$$

$$[\mathsf{get\_answer}(Self, Other, Fluent)]\varphi \equiv \\ [\mathsf{inform}(Other, Self, Fluent) \cup \mathsf{inform}(Other, Self, \neg Fluent) \cup \\ \mathsf{refuseInform}(Other, Self, Fluent)]\varphi$$

Intuitively, the right hand side of get_answer represents all the possible answers expected by agent $Self$ from agent $Other$ about $Fluent$, in the context of a conversation ruled by the $\mathsf{yes\_no\_query}_Q$ protocol.

$$\langle \mathsf{yes\_no\_query}_I(Self, Other, Fluent)\rangle\varphi \subset \\ \langle \mathsf{get\_start}(Self, Other, Fluent); \\ \mathcal{B}^{Self} Fluent?; \mathsf{inform}(Self, Other, Fluent)\rangle\varphi \\ \langle \mathsf{yes\_no\_query}_I(Self, Other, Fluent)\rangle\varphi \subset \\ \langle \mathsf{get\_start}(Self, Other, Fluent); \\ \mathcal{B}^{Self} \neg Fluent?; \mathsf{inform}(Self, Other, \neg Fluent)\rangle\varphi \\ \langle \mathsf{yes\_no\_query}_I(Self, Other, Fluent)\rangle\varphi \subset \\ \langle \mathsf{get\_start}(Self, Other, Fluent); \\ \mathcal{U}^{Self} Fluent?; \mathsf{refuseInform}(Self, Other, Fluent)\rangle\varphi$$

The $\mathsf{yes\_no\_query}_I$ protocol specifies the behavior of the agent $Self$, that waits a query from $Other$; afterwards, it replies according to its beliefs on the query subject. get_start is a get_message action ruled by the following axiom:

$$[\mathsf{get\_start}(Self, Other, Fluent)]\varphi \equiv [\mathsf{queryIf}(Other, Self, Fluent)]\varphi$$

We can define the *communication kit* of an agent ag_i, CKit^{ag_i}, as the triple $(\Pi_{\mathcal{C}}, \Pi_{\mathcal{CP}}, \Pi_{\mathcal{S}get})$, where $\Pi_{\mathcal{C}}$ is the set of simple action laws defining ag_i's primitive speech acts, $\Pi_{\mathcal{S}get}$ is a set of axioms for ag_i's get_message actions and $\Pi_{\mathcal{CP}}$ is the set of procedure axioms specifying the ag_i's conversation protocols. In this extension of the DyLOG language, we define as *Domain Description* for agent ag_i, a triple $(\Pi, \mathsf{CKit}^{ag_i}, S_0)$, where CKit^{ag_i} is ag_i communication kit, S_0 is the

initial set of ag_i's belief fluents, and Π is a tuple $(\Pi_{\mathcal{A}}, \Pi_{\mathcal{S}}, \Pi_{\mathcal{P}})$, where $\Pi_{\mathcal{A}}$ is the set of ag_i's world action and precondition laws, $\Pi_{\mathcal{S}}$ is a set of axioms for ag_i's sensing actions, $\Pi_{\mathcal{P}}$ a set of axioms that define complex actions.

3 Dealing with persistency

We adopt a non-monotonic solution to the persistency problem, by proposing an abductive semantics for our modal language, in which abductive assumptions are used to model persistency of beliefs fluents, from a state to the following one, when an action is performed. The solution is a generalization of the one in [5], so to deal with nested beliefs and communicative actions, and consists in maximizing persistency assumptions about epistemic fluents after the execution of action sequences. In particular we assume that any belief fluent F which holds in a given state persists through an action, unless it is inconsistent to assume so, i.e. unless $\neg F$ holds after the action execution.

Note that belief states are inconsistent when they contain either a belief $\mathcal{B}^{ag_i} l$ and its negation, or the belief formulas $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i} l$ and $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i}\neg l$, or the belief formulas $\mathcal{B}^{ag_j}\mathcal{B}^{ag_i} l$ and $\mathcal{B}^{ag_j}\neg\mathcal{B}^{ag_i} l$. However, from the seriality of the $\mathcal{B}^{ag_i}$ operators, the general formula schema for the rank 2 beliefs

$$\mathcal{B}^{ag_i}\mathcal{B}^{ag_j}\neg\varphi \supset \neg\mathcal{B}^{ag_i}\mathcal{B}^{ag_j}\varphi \tag{7}$$

holds in our logic for any two agents ag_i and ag_j[4]. This property guarantees that when an inconsistency arises "locally" in the beliefs ascribed from ag_i to some other agent, the beliefs of ag_i itself will be inconsistent. Therefore, in case of a nested epistemic fluent $\mathcal{B}^{ag_i}\mathcal{B}^{ag_j} l$, the persistency is *correctly blocked* when a locally inconsistent fluent $\mathcal{B}^{ag_i}\mathcal{B}^{ag_j}\neg l$ becomes true after an action execution, because $\neg\mathcal{B}^{ag_i}\mathcal{B}^{ag_j} l$ can be derived from (7). Given these considerations, we can adopt the same approach to the definition of an abductive semantics, that we followed in [5]. In particular, we adopt the same style used by Eshghi and Kowalski in the definition of the abductive semantics for negation as failure [14]. We define as *abducibles* a new set of atomic propositions of the form $\mathbf{M}[a_1]\dots[a_m]F$.[5] Their meaning is that the fluent expression F can be assumed to hold in the state obtained by the execution of the primitive actions $a_1, \dots, a_m$. Each abducible can be assumed to hold, if it is consistent with the domain description $(\Pi, \mathsf{CKit}^{ag_i}, S_0)$ and with the other assumed abducibles. Then we add to the axiom system, that characterizes the logic defined by the domain description, the *persistency axiom schema*: $[a_1]\dots[a_{m-1}]F \wedge \mathbf{M}[a_1]\dots[a_{m-1}][a_m]F \supset [a_1]\dots[a_{m-1}][a_m]F$, where $a_1, \dots, a_m$ are primitive actions and F is a belief fluent. It means that if F holds after $a_1, \dots, a_{m-1}$, and it can be assumed to persist after action a_m (i.e., it is consistent to assume $\mathbf{M}[a_1]\dots[a_m]F$), then we can conclude that F holds after the

[4] Actually, the general schema for any rank of nesting holds.

[5] Notice that $\mathbf{M}$ is not a modality. $\mathbf{M}\alpha$ denotes a new atomic proposition. $\mathbf{M}\alpha$ means "α is consistent", analogously to default logic.

sequence of actions $a_1, \ldots, a_m$. The definition of abductive solution is given on the line of [5] and is here omitted.

4 Reasoning in presence of communication

Given a domain description, we can reason about it and formalize the *temporal projection* problem and the *planning* problem by existential queries of form:

$$\langle p_1 \rangle \langle p_2 \rangle \ldots \langle p_m \rangle Fs \tag{8}$$

Each p_k, $k = 1, \ldots, m$ in (8) may either be an (atomic or complex) action executed by ag_i or an external speech act, that belongs to CKit^{ag_i} (by the word *external* we denote a speech act in which the agent plays the role of the receiver). By checking if a query of form (8) succeeds we can cope with the planning problem. In fact this corresponds to answering the question "is there an execution trace of $p_1, \ldots, p_n$ leading to a state where the conjunction of belief fluents Fs holds for ag_i?". Such an execution trace is a plan to bring about Fs. The procedure definition constrains the search space. Notice that when all the p_k in the query are atomic actions that belong to $\mathcal{A} \cup \mathcal{C}$, by checking if the query succeeds, we cope also with the temporal projection problem: "does Fs hold for ag_i, after the execution of the action sequence $a_1, \ldots, a_m$?".

In presence of communication, the planning and the temporal projection problems turn respectively into the problem of reasoning about *conversation protocols* and reasoning about simple *conversations*, where a conversation is a sequence of speech acts. This allows, for instance, an agent to *investigate* the possible changes to its mental state, produced by a specific conversation, or if a conversation is an instance of some predefined protocol [13]. In the case of temporal projection, the action sequence will contain both actions in which the agent is the sender and actions in which it is the receiver. In the case of conversation protocols, since they represent conversation schemas that guide the communicative behavior of the agent, by answering to the query (8) we find a conversation, which is an instance of the protocol, after which the desired condition Fs holds. In this process we treat get_message actions as sensing actions, whose outcome cannot be known at planning time. Since agents cannot read each other's mind, they cannot know in advance the answers that they will receive. For this reason all of the possible alternatives are to be taken into account; we can foresee them because of the existence of the protocol. Therefore, the extracted plan will be *conditional*, in the sense that for each get_message and for each sensing action it will contain as many branches as possible action outcomes. Each path in the resulting tree is a linear plan that brings about the desired condition Fs. More formally, a conditional plan σ is either:

- an action sequence $a_1; \ldots; a_m$, with $m \geq 0$;
- if $a_1; \ldots; a_m$ $(m \geq 0)$ is an action sequence, $s \in \mathcal{S}$ is a sensing action, and $\sigma_1, \ldots, \sigma_t$ are conditional plans then $a_1; \ldots; a_m; s; ((\mathcal{B}^{ag_i} l_1?); \sigma_1 \cup \ldots \cup (\mathcal{B}^{ag_i} l_t?); \sigma_t)$, where $l_1, \ldots, l_t \in dom(s)$;

– if $a_1; \ldots; a_m$ ($m \geq 0$) is an action sequence, $g \in \mathcal{S}$ is a get_message action, and $\sigma_1, \ldots, \sigma_t$ are conditional plans then $a_1; \ldots; a_k; g; ((\mathcal{B}^{ag_i} Done(c_1)\top?); \sigma_1 \cup \ldots \cup (\mathcal{B}^{ag_i} Done(c_t)\top?); \sigma_t)$, where $c_1, \ldots, c_t \in \mathcal{C}_g$.

The proof procedure is a natural evolution of the work in [5], it is goal-directed, and based on negation as failure (NAF). NAF is used to deal with the persistency problem to verify that the complement of a mental fluent is not true after an action execution. The proof procedure allows agents to find *linear* and *conditional* plans for achieving a goal from an incompletely specified initial state. The *soundness* w.r.t. the abductive semantics can be proved by imposing domain descriptions to be *e-consistent*, i.e. for any action the set of their effects must be consistent. Moreover, the extracted plans have the following property: they always lead to a state in which the desired condition *Fs* holds, for all the possible results of the sensing actions.

Figure 2 shows the proof procedure that constructs linear plans, by making assumptions on sensing actions and on external communicative actions. Figure 3 introduces a variant for finding conditional plans. In general, we will need to establish if a goal holds at a given state. Hence, we will write:

$$a_1, \ldots, a_m \vdash \langle p_1; p_2; \ldots; p_n \rangle Fs \text{ with answer (w.a.) } \sigma$$

to mean that the query $\langle p_1; p_2; \ldots; p_n \rangle Fs$ can be proved from the domain description at the state $a_1, \ldots, a_m$ with answer σ. σ is an action sequence which represents the state resulting by the execution of $p_1, \ldots, p_n$ in the current state. We denote by ε the empty action sequence that represents the initial mental state. Rules (1–6) in Fig. 2 deal with the execution of complex, sensing, primitive and test actions. The complex actions in the query are reduced to a sequence of primitive and test actions; the proof procedure verifies if the primitive actions can be executed and if the tests are successful. To do this, it reasons about the execution of the primitive actions and computes the values of fluents at different states. The value of fluents at a state is not explicitly recorded but it is computed when needed in the computation. Rules (7–14), allow the values of mental fluents to be determined and, in particular, to determine if *Fs* is true after $a_1, \ldots, a_m$. An epistemic fluent F holds in the current state if: either F is an immediate effect of action a_m, whose preconditions hold in the previous state (8a); or a_m is an *ad hoc* primitive action, used in the definition of a sensing action (8b); or F persists from the previous state (8c); or we are in the initial state and F holds (8d). Rule (8c) deals with the *frame problem*: F persists from a state to the next one unless a_m makes $\neg F$ true; **not** represents NAF. Rules (10-12) deal with the seriality (10), transitivity (11 and 11'), and euclideanity (12 and 12') of the beliefs. Rules (13) and (14) deal with the simmetry and awareness of action's execution. Fig. 3 reports the two rules that substitute (4) and (5) in Fig. 2 to build conditional plans. The new rules deal with the execution of sensing and get message actions, respectively. As a difference with the previous proof procedure, when a sensing action is executed, the procedure considers *all the possible outcomes*, thus producing many branches. If all branches lead to success, the main

1) $\dfrac{\overline{a}_{1\cdots m} \vdash \langle p'_1; \ldots; p'_{n'}; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}{\overline{a}_{1\cdots m} \vdash \langle p; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}$ where $p \in \mathcal{P}$ and $\langle p\rangle\varphi \subset \langle p'_1; \ldots; p'_{n'}\rangle\varphi \in \Pi_{\mathcal{P}} \cup \Pi_{\mathcal{CP}}$

2) $\dfrac{\overline{a}_{1\cdots m} \vdash Fs' \quad \overline{a}_{1\cdots m} \vdash \langle \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}{\overline{a}_{1\cdots m} \vdash \langle (Fs')?; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}$

3) $\dfrac{\overline{a}_{1\cdots m} \vdash Fs' \quad \overline{a}_{1\cdots m}, a \vdash \langle \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}{\overline{a}_{1\cdots m} \vdash \langle a; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}$ where $a \in \mathcal{A} \cup \mathcal{C}$, and $\Box(Fs' \supset \langle a\rangle\top) \in \Pi_{\mathcal{A}} \cup \Pi_{\mathcal{C}}$

4) $\dfrac{\overline{a}_{1\cdots m} \vdash \langle s^{\mathcal{B}^{ag_i} l}; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}{\overline{a}_{1\cdots m} \vdash \langle s; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}$ where $s \in \mathcal{S}$ and $l \in dom(s)$

5) $\dfrac{\overline{a}_{1\cdots m} \vdash \langle c; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}{\overline{a}_{1\cdots m} \vdash \langle g; \overline{p}_{2\ldots n}\rangle Fs \text{ w. a. } \sigma}$ where $g \in \mathcal{S}get$ and $[g]\varphi \equiv [\bigcup_{c\in\mathcal{C}_g} c]\varphi$

6) $\dfrac{\overline{a}_{1\cdots m} \vdash Fs}{\overline{a}_{1\cdots m} \vdash \langle\varepsilon\rangle Fs \text{ w. a. } \sigma}$ where $\sigma = a_1; \ldots; a_m$

7) $\dfrac{}{\overline{a}_{1\cdots m} \vdash \top}$

8a) $\dfrac{\overline{a}_{1\cdots m-1} \vdash Fs'}{\overline{a}_{1\cdots m} \vdash F}$ where $m > 0$ and $\Box(Fs' \supset [a_m]F) \in \Pi_{\mathcal{A}}$

8b) $\dfrac{}{\overline{a}_{1\cdots m} \vdash F}$ if $a_m = s^F$

8c) $\dfrac{\textbf{not } \overline{a}_{1\cdots m} \vdash \neg F \quad \overline{a}_{1\cdots m-1} \vdash F}{\overline{a}_{1\cdots m} \vdash F}$ where $m > 0$

8d) $\dfrac{}{\varepsilon \vdash F}$ if $F \in S_0$

9) $\dfrac{\overline{a}_{1\cdots m} \vdash Fs' \quad \overline{a}_{1\cdots m} \vdash Fs''}{\overline{a}_{1\cdots m} \vdash Fs' \wedge Fs''}$

10) $\dfrac{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i} L}{\overline{a}_{1\cdots m} \vdash \mathcal{M}^{ag_i} L}$

11) $\dfrac{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i} l}{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i}\mathcal{B}^{ag_i} l}$ 11') $\dfrac{\overline{a}_{1\cdots m} \vdash \mathcal{M}^{ag_i}\mathcal{M}^{ag_i} l}{\overline{a}_{1\cdots m} \vdash \mathcal{M}^{ag_i} l}$

12) $\dfrac{\overline{a}_{1\cdots m} \vdash \mathcal{M}^{ag_i} l}{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i}\mathcal{M}^{ag_i} l}$ 12') $\dfrac{\overline{a}_{1\cdots m} \vdash \mathcal{M}^{ag_i}\mathcal{B}^{ag_i} l}{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i} l}$

13) $\dfrac{\overline{a}_{1\cdots m} \vdash Done(a)\top}{\overline{a}_{1\cdots m} \vdash \mathcal{B}^{ag_i} Done(a)\top}$ 14) $\dfrac{}{\overline{a}_{1\cdots m} \vdash Done(a_m)\top}$

Fig. 2. A goal directed proof procedure for DyLOG. Legend: $\overline{a}_{1\cdots m} \equiv a_1, \ldots, a_m$ and $\overline{p}_{2\ldots n} \equiv p_2, \ldots, p_n$. l denotes a fluent literal or a done fluent while L denotes l or a belief fluent of rank 1.

$$\text{4-bis)}\quad \frac{\forall l_k \in \mathcal{F},\ \overline{a}_{1\cdots m} \vdash \langle s^{\mathcal{B}^{ag_i} l}; \overline{p}_{2\cdots n}\rangle Fs \text{ w. a. } a_1;\ldots;a_m; s^{\mathcal{B}^{ag_i} l}; \sigma'_k}{\overline{a}_{1\cdots m} \vdash \langle s; \overline{p}_{2\cdots n}\rangle Fs \text{ w. a. } a_1;\ldots;a_m; s; (\bigcup_{k=1\ldots t}(\mathcal{B}^{ag_i} l_k?); \sigma'_k)}$$

$$\text{5-bis)}\quad \frac{\forall c_k \in \mathcal{C}_g,\ \overline{a}_{1\cdots m} \vdash \langle c_k; \overline{p}_{2\cdots n}\rangle Fs_i \text{ w. a. } a_1;\ldots;a_m; c_k; \sigma'_k}{\overline{a}_{1\cdots m} \vdash \langle g; \overline{p}_{2\cdots n}\rangle Fs_i \text{ w. a. } a_1;\ldots;a_m; g; (\bigcup_{k=1\ldots t}(\mathcal{B}^{ag_i} Done(c_k)\top?); \sigma'_k)}$$

Fig. 3. A variant of the proof procedure for extracting conditional plans. In (4-bis) $s \in \mathcal{S}$ and $\mathcal{F} = \{l_1, \ldots, l_t\} = dom(s)$; in (5-bis) $g \in \mathcal{S}$ and $\{c_1, \ldots, c_t\} = \mathcal{C}_g$.

query succeeds. In such a case, the conditional plan will contain the branches as alternative sub-plans. The same holds for the execution of get_message actions.

Example 1. Let us consider the protocol $\mathsf{get\_ticket\_1}_C$, describing from the *customer* perspective the interaction with a *cinema booking service*:

(a) $\langle \mathsf{get\_ticket\_1}_C(Self, WebS, Film)\rangle\varphi \subset$
$\langle \mathsf{yes\_no\_query}_Q(Self, WebS, available(Film))\ ;\ \mathcal{B}^{Self} available(Film)?\ ;$
$\mathsf{get\_info}(Self, WebS, cinema(C))\ ;\ \mathsf{yes\_no\_query}_I(Self, WebS, pay\_by(credit\_card))\ ;$
$\mathcal{B}^{Self} pay\_by(credit\_card)?\ ;\ \mathsf{inform}(Self, WebS, cc\_number);$
$\mathsf{get\_info}(Self, WebS, booked(Film))\rangle\varphi$

(b) $\langle \mathsf{get\_ticket\_1}_C(Self, WebS, Film)\rangle\varphi \subset$
$\langle \mathsf{yes\_no\_query}_Q(Self, WebS, available(Film))\ ;\ \mathcal{B}^{Self} available(Film)?\ ;$
$\mathsf{get\_info}(Self, WebS, cinema(C))\ ;\ \mathsf{yes\_no\_query}_I(Self, WebS, pay\_by(credit\_card));$
$\neg\mathcal{B}^{Self} pay\_by(credit\_card)?\ ;\mathsf{get\_info}(Self, WebS, pay\_by(cash));$
$\mathsf{get\_info}(Self, WebS, booked(Film))\rangle\varphi$

(c) $\langle \mathsf{get\_ticket\_1}_C(Self, WebS, Film)\rangle\varphi \subset$
$\langle \mathsf{yes\_no\_query}_Q(Self, WebS, available(Film)); \neg\mathcal{B}^{Self} available(Film)?\rangle\varphi$

(d) $[\mathsf{get\_info}(Self, WebS, Fluent)]\varphi \equiv [\mathsf{inform}(WebS, Self, Fluent)]\varphi$

$\mathsf{get\_ticket\_1}_C$ permits both to book a ticket to be paid later by cash and to buy it by credit card; suppose it is followed by the web service *click_ticket*. Given the query $\langle \mathsf{get\_ticket\_1}_C(pa, click\_ticket, akira)\rangle\mathcal{B}^{pa}\neg\mathcal{B}^{click\_ticket} cc\_number$, a *personal assistant pa* could reason on it to determine if there is a conversation between *pa* and *click_ticket* about the movie *akira*, after which the service does *not* know the credit card number of the user. Since such a conversation exists, the agent *pa* finds an execution trace of $\mathsf{get\_ticket\_1}_C$, which corresponds to a *personalized conditional dialogue plan* between itself and the provider *click_ticket*, always leading to satisfy the user goal of not giving the credit card number. For a deeper discussion about personalization of web service fruition see [3].

5 Conclusion and related work

Communication among agents has extensively been studied by the AI community. One of the most popular approaches, derived from the work of philosophers and linguists carried on in the sixties [2, 23], considers *rationality* as a key concept. In other words, communicative acts are interpreted as rational actions with

preconditions and effects on the agent mental state, that can be planned and reasoned about [11, 1, 10]; this approach lead to the definition of well-known ACLs like FIPA [17]. The semantics of communication can be given at different levels of detail. In many formal approaches [11, 10, 6, 18] the focus is posed at the level of the single speech acts and the task of reasoning about communication and planning is achieved based on their preconditions and desired effects, without considering them in the context of a conversation protocol. Indeed many of these approaches [6, 18] have been born for the developement of intelligent human-machine dialogue systems, then they are focussed on techniques where recognizing intentions in communications is fundamental for producing a suitable reply. On the line of [20], we argue that the use of conversation protocols makes the design of software components that must interact easier: the interoperability of the various components (often separately developed) is improved and the verification of compliance to the desired standards is simplified. By working at the level of protocols, agents can more easily be seen as individuals, developed independently, on different platforms and with different approaches, a very attractive view in the applicative field of web applications and web services. For all these reasons we focus on a semantics of communication that supports the specification and reasoning about single speech acts, as well as the specification and reasoning about speech acts in the context of a conversation protocol. In our framework, protocols are intended as tractable decision procedures, that the agent can use for selecting and producing communicative acts, suitable to the agent goals. Since they limit the domain of possible interactions, an advantage is that they reduce the search space.

More specifically, we have presented an approach to reason about conversation protocols within the framework of an agent language based on a modal logic of action and beliefs. The approach extends with communication the proposal to model rational agents in [5]. We used conversation protocols to provide our agents decision procedures for suitably responding to communications. We took a subjective representation of conversation protocols, by making hypothetical assumptions on the other's answers. As a consequence protocols have been easily integrated with other policies defining the agent's behavior, being both represented as procedures specified in DyLOG. Notice that, since we are only interested in reasoning about the local mental state's dynamics, our approach differs from other logic-based approaches to communication in multi-agent systems, as the one taken in [24], where communicative actions affect the global state of a multi-agent system and the target is to prove global properties of the overall multi-agent system's execution. Instead our focus on the internal specification of interaction protocols for planning dialogue's moves is closer to the one taken in [22], where negotiation protocols, expressed by sets of dialogue constraints, are included in the agent program and used for triggering dialogues that achieve goals. However such an approach is not aimed at implementing the kind of reasoning about conversations we focused on: it does not support plan extraction and it cannot exploit the information about the others, that instead we can supply by nested beliefs.

References

1. J. F. Allen. Recognizing intentions from natural language utterances. In M. Brady and R.C. Åqvist, editors, *Computational models of discourse*. MIT Press, Cambridge, MA, 1983.
2. J.A Austin. *How to do things with words*. Harvard University Press, 1962.
3. M. Baldoni, C. Baroglio, A. Martelli, and V. Patti. Reasoning about interaction for personalizing web service fruition. In *Proc. of WOA 2003: dagli Oggetti agli Agenti*, Cagliari, Italy, september 2003. to appear.
4. M. Baldoni, C. Baroglio, V. Patti, and L. Torasso. Using a rational agent in an adaptive web-based tutoring system. In *Proc. of the Workshop on Adaptive Systems for Web-Based Education, AH2002*, Malaga, Spain, 2002.
5. M. Baldoni, L. Giordano, A. Martelli, and V. Patti. Reasoning about Complex Actions with Incomplete Knowledge: A Modal Approach. In *Proc. of ICTCS'2001*, volume 2202 of *LNCS*, pages 405–425. Springer, 2001.
6. P. Bretier and D. Sadek. A rational agent as the kernel of a cooperative spoken dialogue system: implementing a logical theory of interaction. In *Intelligent Agents III, proc. of ECAI-96 Workshop on Agent Theories, Architectures, and Languages (ATAL-96)*, *LNAI* 1193. Springer-Verlag, 1997.
7. J. Bryson, D. Martin, S. McIlraith, and L. A. Stein. Agent-based composite services in DAML-S: The behavior-oriented design of an intelligent semantic web, 2002.
8. R. Chinnici, M. Gudgin, J. J. Moreau, and S. Weerawarana. Web Services Prescription Language (WSDL) version 1.2, 2003. Working Draft.
9. The DAML-S coalition. DAML-S: Web service description for the semantic web. In *the 1st Int. Semantic Web Conference (ISWC)*, Sardinia, Italy, 2002.
10. P.R. Cohen and H. Levesque. Rational interaction as the basis for communication. In P.R Cohen, M.E. Pollack, and J. Morgan, editors, *Intentions in Communication*, pages 221–256, 1990.
11. P.R. Cohen and C.R. Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3):177–212, 1979.
12. M. Dastani, J. van der Ham, and F. Dignum. Communication for goal directed agents. In *Proc. of Workshop on Agent Communication Languages and Conversation Policies, AAMAS'02*, Bologna, Italy, 2002.
13. F. Dignum and M. Greaves. Issues in agent communication. In *Issues in Agent Communication*, volume 1916 of *LNCS*, pages 1–16. Springer, 2000.
14. K. Eshghi and R. Kowalski. Abduction compared with Negation by Failure. In *Proc. of ICLP '89*, Lisbon, 1989. The MIT Press.
15. T. Finin, Y. Labrou, and J. Mayfield. KQML as an Agent Communication Language. In J. Bradshaw, editor, *Software Agents*. MIT Press, 1995.
16. FIPA. FIPA 2000. Technical report, FIPA (Foundation for Intelligent Physical Agents), November 2000.
17. FIPA. FIPA 2002. Technical report, FIPA (Foundation for Intelligent Physical Agents), 2002.
18. A. Herzig and D. Longin. Beliefs dynamics in cooperative dialogues. In *Proc. of AMSTELOGUE 99*, 1999.
19. H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. B. Scherl. GOLOG: A Logic Programming Language for Dynamic Domains. *J. of Logic Programming*, 31:59–83, 1997.
20. A. Mamdani and J. Pitt. Communication protocols in multi-agent systems: A development method and reference architecture. In *Issues in Agent Communication*, volume 1916 of *LNCS*, pages 160–177. Springer, 2000.

21. J. H. Odell, H. Van Dyke Parunak, and B. Bauer. Representing agent interaction protocols in UML. In *Agent-Oriented Software Engineering*, pages 121–140. Springer, 2001.
22. F. Sadri, F. Toni, and P. Torroni. Dialogues for Negotiation: Agent Varieties and Dialogue Sequences. In *Proc. of ATAL'01*, Seattle, WA, 2001.
23. J.R. Searle. *Speech Acts*. Cambridge University Press, New York, 1969.
24. S. Shapiro, Y. Lespérance, and H. J. Levesque. Specifying communicative multi-agent systems. In *Agents and Multi-Agent Systems - Formalisms, Methodologies, and Applications*, volume 1441 of *LNAI*, pages 1–14. Springer-Verlag, 1998.