Jamal Bentahar

A Pragmatic and Semantic Unified Framework for Agent Communication

Thèse présentée
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de doctorat en Informatique
pour l'obtention du grade de Philosophiæ Doctor (Ph.D.)

Faculté des Sciences et de Génie
Université Laval
Québec

Avril, 2005

Abstract

In this thesis, we propose a unified framework for the pragmatics and the semantics of agent communication. Pragmatics deals with the way agents use communicative acts when conversing. It is related to the dynamics of agent interactions and to the way of connecting individual acts while building complete conversations. Semantics is interested in the meaning of these acts. It lays down the foundation for a concise and unambiguous meaning of agent messages. This framework aims at solving three main problems of agent communication:

1- The absence of a link between the pragmatics and the semantics.
2- The inflexibility of current agent communication protocols.
3- The verification of agent communication mechanisms.

The main contributions of this thesis are:

1- A formal pragmatic approach based on social commitments and arguments.
2- A new agent communication formalism called Commitment and Argument Network.
3- A logical model defining the semantics of the elements used in the pragmatic approach.
4- A tableau-based model checking technique for the verification of a kind of flexible protocols called dialogue game protocols.
5- A new persuasion dialogue game protocol.

The main idea of our pragmatic approach is that agent communication is considered as actions that agents perform on social commitments and arguments. The dynamics of agent conversation is represented by this notion of actions and by the evolution of these commitments and arguments. Our Commitment and Argument Network formalism based on this approach provides an external representation of agent communication dynamics. We argue that this formalism helps agents to participate in conversations in a flexible way because they can reason about their communicative acts using their argumentation systems and the current state of the conversation.

Our logical model is a model-theoretic semantics for the pragmatic approach. It defines the meaning of the different communicative acts that we use in our pragmatic approach. It also expresses the meaning of some important speech acts and it captures the semantics of defeasible arguments. This logical model allows us to establish the link between the semantics and the pragmatics of agent communication.

We address the problem of verifying dialogue game protocols using a tableau-based model checking technique. These protocols are specified in terms of our logical model. We argue that our model checking algorithm provides a technique, not only to verify if the dialogue game protocol satisfies a given property, but also if this protocol respects the underlying semantics of the communicative acts.

Our persuasion dialogue game protocol is specified in our framework using a logical language, and implemented using a logic programming and agent-oriented programming paradigm. In this protocol, the agents' decision making process is based on the agents' argumentation systems and the notion of agents' trustworthiness.

Résumé

Dans cette thèse, nous proposons un cadre unifié pour la pragmatique et la sémantique de la communication entre agents logiciels. La pragmatique traite la façon dont les agents utilisent les actes communicatifs lorsqu'ils participent aux conversations. Elle est liée à la dynamique des interactions entre agents et à la manière avec laquelle les actes individuels sont reliés pour construire des conversations complètes. La sémantique, quant à elle, est intéressée par la signification de ces actes. Elle établit la base pour une signification concise et non ambiguë des messages échangés entre les agents. Ce cadre unifié vise à résoudre trois problèmes majeurs dans le domaine de communication entre agents :

1- L'absence d'un lien entre la pragmatique et la sémantique.
2- L'inflexibilité des protocoles actuels de communication entre agents.
3- La vérification des mécanismes de communication entre agents.

Les contributions principales de cette thèse sont :

1- Une approche pragmatique formelle basée sur les engagements sociaux et les arguments.
2- Un nouveau formalisme pour la communication entre agents appelé Réseau d'Engagements et d'Arguments.
3- Un modèle logique définissant la sémantique des éléments utilisés dans l'approche pragmatique.
4- Une technique de vérification de modèles basée sur une sémantique à tableaux pour vérifier une famille de protocoles flexibles de communication entre agents appelée protocoles à base de jeux de dialogue.
5- Un nouveau protocole de persuasion à base de jeux de dialogue.

L'idée principale de notre approche pragmatique est que la communication entre agents est modélisée comme des actions que les agents accomplissent sur des engagements sociaux et des arguments. La dynamique de la conversation entre agents est représentée par cette notion d'actions et par l'évolution de ces engagements et arguments. Notre formalisme (Réseau d'Engagements et d'Arguments) basé sur cette approche fournit une représentation externe de la dynamique de communication entre agents. Ce formalisme peut être utilisé par les agents comme moyen pour participer à des conversations d'une manière flexible parce qu'ils peuvent raisonner sur leurs actes communicatifs en utilisant leurs systèmes d'argumentation et l'état actuel de la conversation.

Notre modèle logique est une sémantique, à base d'un modèle théorique, pour l'approche pragmatique. Il définit la signification des différents actes de communication que nous utilisons dans notre approche pragmatique. Il exprime également la signification de quelques actes de discours importants dans le contexte de communication multi-agents et il

capture la sémantique des arguments annulables. Ce modèle logique permet d'établir le lien entre la sémantique et la pragmatique de communication entre agents.

Nous traitons le problème de vérification des protocoles à base de jeux de dialogue en utilisant une technique de vérification de modèles basée sur une sémantique à tableaux. Ces protocoles sont spécifiés sur la base de notre modèle logique. Nous montrons que notre algorithme de vérification offre une technique, non seulement pour vérifier si le protocole à base de jeux de dialogue (le modèle) satisfait une propriété donnée, mais également si ce protocole respecte la sémantique des actes communicatifs.

Notre protocole de persuasion à base de jeux de dialogue est spécifié dans le contexte de notre cadre unifié en utilisant un langage logique. Il est implémenté en utilisant une programmation logique et un paradigme orienté-agent. Dans ce protocole, le processus décisionnel des agents est basé sur les systèmes d'argumentation et sur la notion de crédibilité des agents.

Acknowledgments

First, I would like to warmly thank my thesis supervisor, Bernard Moulin, for his invaluable support and patience, for his precious and never lacking enthusiasm for research and for his continuous assistance in preparing and writing this thesis. I also want to thank him for having so strongly believed in me and provided me with insights which helped me solve many of the problems I encountered in my research. I also acknowledge him for assisting me financially during my Ph.D and for his generosity which gave me opportunities to attend several international conferences and to visit Utrecht University in order to validate my work with the members of Intelligent Systems Group.

I would like to express my gratitude to my co-supervisor, Brahim Chaib-draa, whose expertise, understanding and patience, added considerably to my graduate experience. I want to thank him for highly stimulating discussions (which generally took place in the department corridor) that have led to the discovery of many results in this thesis and for having given me the opportunity to collaborate with many researchers from Utrecht University, Université Paris 9 Dauphine and University of Melbourne.

I want to thank Yves Lespérance from York University (Canada) for having accepted to evaluate the first version of this thesis. I would like to thank him for his very interesting comments and suggestions that allowed me to improve the quality of this thesis. I appreciated the many discussions that I have had with him about the different parts of this dissertation.

I also wish to gratefully thank Frank Dignum from Utrecht University (the Netherlands) and Josée Desharnais from Laval University (Canada) for having accepted to evaluate my thesis and participate in the jury.

A very special thanks goes out to John-Jules Ch. Meyer from Utrecht University (the Netherlands) for his extremely appreciated collaboration that allowed me to improve the logical model and the model checking technique proposed in this thesis. I have very much enjoyed working with him. I also would like to thank him for having received me in his Intelligent Systems Group for 2 weeks during the last month of my thesis and for having given me the opportunity to discuss my work with Frank Dignum, Mehdi Dastani, Henry Prakken, Broersen Jan, Martin Caminada, Van Riemsdijk Birna and Davide Grossi. Particular thanks goes out to Frank Dignum for his helpful comments and interesting suggestions.

I am particularly appreciative of the help I did receive from: Leila Amgoud from Institut de Recherche en Informatique de Toulouse (France), Nicolas Maudet from Université Paris 9 Dauphine (France), Marco Colombetti from Politecnico di Milano (Italy), Iyad Rahwan from University of Melbourne (Australia) and British University in Dubai (United Arab Emirates), Rance Cleaveland from State University of New York at Stony Brook (USA), Girish Bhat from Cosine Communication Inc. (USA), Peter McBurney from University of Liverpool (UK), Simon Parsons from City University of New York (USA), Claude Bélisle from Université Laval, Department of Mathematics and Statistics (Canada), Josée Desharnais from Université Laval (Canada), Daniel Vanderveken from Université de Québec à Trois-Rivières, Department of Philosophy (Canada), Sylvain Delisle from Université de Québec à Trois-Rivières (Canada) and my colleague Philippe Pasquier. I also would like to thank Johanne Savard for her friendship and linguistic help.

My years as a Ph.D student would not have been as much fun without my friends, particularly my colleagues: Nafaa Jabeur, Nabil Sahli, Walid Ali, Hedi Haddad, Mondher Bouden, Tarek Sbeoui, and Boulekrouche Boubaker with whom I shared the laboratory and unforgettable moments. I would like to thank him for being the surrogate family and for their continued moral support. From the staff, Linda Goulet and Violaine Pellerin are especially thanked for their care and attention.

Finally, I would like to thank my mother and all my family (Fatima, Mohammed, Faisal, Hassan, Fadila, Bouchra, and Wiam) for the support they provided me through my entire life and for their endless patience, without whose love, encouragement and editing assistance, I would not have finished this thesis.

To my mother and in memory of my father,
to my family
with love and gratitude

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we introduce the context of our research which is communication between software agents in a multi-agent system. We also identify the motivations, problems, and research questions that we address in this thesis. Finally, we present our hypotheses, objectives, and methodology.

1.1 Context of the Research

This thesis is about communication between autonomous agents. In the multi-agent domain, it is widely recognized that this communication is one of the major topics of research. All the applications of Multi-Agent Systems (MASs) (Chaib-draa, 1995), (Wooldridge and Jennings, 1995), (Moulin and Chaib-draa, 1996) (Wooldridge, 2002) ranging from digital libraries through cooperative engineering to electronic commerce, have one thing in common: the agents operating in these systems have to communicate. These systems consist of multiple agents that communicate in order to solve some problems. If a problem is particularly complex, large, or unpredictable, the only way it can reasonably be addressed is to develop a number of functionally specific and modular components (agents) which are able to solve a particular problem aspect (Sycara, 1998). This decomposition allows each agent to use the most appropriate paradigm to solve its particular problem. When interdependent problems arise, agents in the system must communicate in order to coordinate with one another to ensure that interdependencies are properly managed. Therefore, it is clear that the success of these systems need powerful communication mechanisms.

Agent communication is related to several disciplines: philosophy of language, social psychology, artificial intelligence, logic, mathematics, etc. In this domain, in order to be able to negotiate, solve conflicts of interest, cooperate, find proofs, agents need not only exchange single messages, but also take part in conversations with other agents. A conversation is defined as a coherent sequence of utterances. The term “coherent” means that the information conveyed by an utterance is related to the information conveyed by the other utterances in a conversation. For example, if p is the information conveyed by an utterance, the information conveyed by the next one can be the acceptance, the refusal, the challenge, the attack, etc. of p. Indeed, if agents communicate by exchanging isolated messages, the resulting communication is extremely poor and agents cannot participate in complex interactions such as negotiations, persuasions, deliberations, etc, which are formed by a sequence of utterances.

The language used by the agents for their exchanges is the Agent Communication Language (ACL). An ACL stems from the need to coordinate the actions of an agent with that of the other agents. A first attempt to come to a standardized ACL came from the DARPA knowledge sharing project and produced KQML (Knowledge Query and Manipulation Language) (Finin et al., 1995). Another effort to come to a standard ACL has started through the Foundation for Intelligent Physical Agents (FIPA) initiative (FIPA, 2001a, 2002). KQML and FIPA-ACL are based on speech act theory and messages are considered as communicative acts whose objective is to perform some action by virtue of being sent. To enable agents to communicate, FIPA proposed a set of communication protocols that agents can follow. FIPA contract net interaction protocol is an example of these protocols (FIPA Interaction Protocols, 2001, 2002). In the contract net interaction protocol, one agent (the initiator) takes the role of manager which wishes to have some task performed by one or more other agents (the participants) and further wishes to optimise a function that characterizes the task. This characteristic is commonly expressed as the price, in some domain specific way, but could also be soonest time to completion, fair distribution of tasks, etc. Generally, agent communication protocols describe the sequence of messages that agents can exchange for particular applications. Although these protocols can successfully be used for some simple applications, they are often too rigid to be used by autonomous agents in their conversations. The reason is that these protocols are specified in such a way that agents must follow them from beginning to end without specifying how these agents can reason about them. To solve this problem, several researchers proposed dialogue game frameworks inspired by the philosophy of argumentation (Reed, 1998), (McBurney and Parsons, 2001, 2002), (Dignum et al., 2000, 2001). Dialogue games are abstract structures that can be composed in order to reflect the whole dialogue. They are interactions between two or more players, in which each player moves by making utterances, according to a pre-defined set of rules. The rules typically define what locutions may or must be uttered in different circumstances. However, the underlying semantics and the verification of these dialogue games are aspects yet to be addressed.

1.2 Motivations

To be able to communicate, agents should use a common communication mechanism (for example, a communication protocol). Because agents are autonomous, this mechanism must be flexible enough and agents must reason about their communicative acts in order to decide how they can pursue their conversations. Classical protocols, like those used in distributed systems, are not suitable in this domain because they only describe the sequence of allowed actions without any reasoning mechanism. Our first motivation is to give agents flexible means of communication. These means must be formally specified by taking into account the agents' architecture.

In addition, in the domain of agent communication, semantics is one of the most important aspects, particularly in the current context of open and interoperable MASs. Semantics lays down the foundation for a concise and unambiguous meaning of agent messages. When agents interact to achieve a goal, the mutual understanding of the exchanged messages depends on the semantics given to communicative actions. Although some significant research work was done in this field (Singh, 2000) (Guerin and Pitt, 2001) (Amgoud et al., 2002), (McBurney, 2002), (Verdicchio and Colombetti, 2003), (Flores et al., 2004) the definition of a clear and global semantics is an objective yet to be reached. Agent communication pragmatics is another important aspect to be addressed in this domain. While semantics is interested in the meaning of communication acts, pragmatics deals with the way of using these acts. Pragmatics is related to the dynamics of agent communication. Because agents do not exchange isolated messages, but participate in complete conversations, the semantics must take into account the chaining of the communicative actions. Therefore, the semantics must be defined in a pragmatic perspective. Our second motivation is to contribute to the advance of research in this domain by developing a unified formal framework establishing the link between the pragmatics and semantics of agent communication. This motivation is related to the first one in the sense that the formal specification of the communication mechanism should allow us to verify whether or not agents respect the defined semantics when conversing.

1.3 Problems and Research Questions

The first problem that we address in this thesis is the lack of flexibility in most current agent communication protocols (FIPA Interaction Protocols, 2001, 2002). These protocols are static and agents must execute them from beginning to end in order to communicate. In addition, these protocols do not specify how agents can manage exceptions (messages not specified or not supported by the protocol), and how they can choose a communicative act among others. To address this problem, several researchers proposed dialogue game frameworks. These frameworks attempt to support more complex conversations by combining different atomic dialogues. Agents participating in a dialogue game framework must agree on all the rules of the framework. However, several proposals in this domain do not specify how agents can reason about these rules and participate in conversations in a flexible way. If the decision making process belongs to the agent architecture, the link between this architecture and the communication model must be specified. In addition, these frameworks do not take into account the link between the private mental states and the reasoning abilities of agents. Thus, our initial research question is: "*How may autonomous agents participate in conversations in a flexible way?*"

In the literature, three main approaches have been proposed for modeling agent communication: the mental approach, the social approach, and the argumentative approach. The mental approach focuses on the agents' private mental states like beliefs, desires and intentions. In this approach, the semantics of the communicative acts is defined using these mental states. The social approach highlights the public and observable elements like social commitments that agents exchange when conversing. A social semantics is defined using the notion of social commitment. The argumentative approach is based on the agents' reasoning capabilities. The meaning of communicative acts in this approach is defined in terms of arguments in favor or against the content of these acts. These approaches reflect only a partial view of agent communication. When participating in conversations, agents should use their mental states, exchange observable elements, and reason about these states and elements. Therefore, these approaches should be combined in a unified approach. The question that we explore here is: "*How can we unify these approaches?*" Another related question is: "*How can the link between pragmatics and semantics be established in such an approach?*"

The third problem that we explore in this thesis is the verification of agent communication. Two fundamental aspects need to be verified when specifying and developing agent communication mechanisms: the agents' compliance to the ACL semantics, and the correctness of the specification in the sense that the mechanism satisfies some given properties. Although this verification is extremely important in open environment and in complex and interoperable systems, the different protocols proposed in the literature (classical or based on dialogue games) do not address it. Verifying these protocols is not an easy task when considering the different states of agents and their reasoning capabilities. The question is: "*How can we formally specify and verify the agent communication mechanisms?*" In addition, the termination of agent conversations and the complexity analysis of the corresponding reasoning algorithms must be addressed.

1.4 Research Hypotheses

To take part in flexible conversations (persuasions, argumentative negotiations, deliberations, etc.), software agents must have a suitable communication model. Agents must build their conversation dynamically while it advances. Thus, our first research hypothesis is that in their conversations, agents do not have to follow pre-established and fixed protocols. Instead, they need to reason about all utterances that have been uttered during the conversation in order to decide about what is necessary to utter next. In flexible conversations, protocols are only interesting as long as agents can use them as stereotypes which can help them in their conversations and not as means imposing what agents must do. Protocols only specify the allowed communicative acts, and do not indicate how agents can choose between these acts. In other words, protocols do not specify the underlying decision making process which is fundamental for conversing agents.

The second hypothesis is related to the importance of the conversation context. The conversation context is defined by the set of knowledge and beliefs that agents suppose they share during their conversations. For example, as members of the same cultural community, the participants in a conversation share knowledge of a general nature and knowledge related to the existing standards and conventions. We make the hypothesis that agents communicate in a particular context that they share.

1.5 Objectives

The main objectives of this thesis are:

1- To propose a pragmatic approach for agent communication taking into account the different elements that agents use in their conversations. This pragmatic approach based on social commitments and arguments must illustrate how agents use their communicative acts when conversing. It must also represent the dynamics of agent communication. This approach, based on speech act theory and specified by a formal language, will be used to develop a formal framework allowing agents to take part in conversations in a flexible way.

This framework, specified as a mathematical structure, should be able to represent the various actions that can take place in agent conversations and to model the dynamics of these conversations.

2- To develop a communication model and a corresponding agent architecture on the basis of the pragmatic approach.

3- To define a formal semantics related to the pragmatic approach. The idea is to specify a unified framework for the pragmatics and the semantics of agent communication. The meaning of the communicative acts must take into account the dynamics of agent communication.

4- To develop a verification method for dialogue game protocols specified using the unified framework.

5- To specify and to implement a flexible dialogue game protocol using the unified framework.

1.6 Methodology

At the beginning of this thesis, we studied research work done in the domain of agent communication. We noticed that the classical protocols used in this field are not suitable in the context of MASs in which agents are autonomous. Particularly, we noted the absence of the reasoning aspect in these protocols. For this reason, we looked at the work done in another field: argumentation and defeasible reasoning. We had the idea to combine an approach proposed in the domain of agent communication, the social approach, which has the advantage of being semantically verifiable and an argumentative approach.

In addition, we noticed that the traditional formalisms used to model agent communication are limited. They do not make it possible to reflect the dynamics of this communication in terms of the actions which agents perform when conversing and do not help agents to take part in these conversations in a flexible way. We thus developed a formalism addressing these limits using our hybrid approach. The proposed approach and formalism only reflect the pragmatics of agent communication. To deal with the semantic aspect, we proposed a logical model for the pragmatic approach. Indeed, we developed a unified framework for the pragmatics and semantics of agent communication.

Although certain researchers recently started to emphasize the importance of verifying MASs, this aspect has yet to be explored in the field of agent communication. In this domain, only a small amount of research work addressed this complicated issue, for example (Wooldridge, 2000) (Huget and Wooldridge, 2004). For this reason, we studied more profoundly this aspect which is traditionally related to software engineering. We proposed a model-checking method in order to verify dialogue game protocols specified using our unified framework. Finally, as an application of our theoretical results, we specified and implemented a flexible dialogue game protocol. We proved its termination and we discussed its computational complexity.

1.7 Overview of the Dissertation

This thesis is divided into two parts.

Part I is about the state of the art, and it consists of three chapters:

> **Chapter 2** introduces the agent communication. In this chapter we present some examples of Agent Communication Languages (ACLs), we discuss their semantics, and present their philosophical foundations.
>
> **Chapter 3** presents some dialogue game frameworks. In this chapter, we highlight their theoretical foundations, advantages, and limits. These limits will be addressed in our proposal.
>
> **Chapter 4** presents our taxonomy of the main approaches in the domain of agent communication and dialogue modeling. This chapter compares and discusses these different approaches.

Part II consists of five chapters in which we present our contributions:

> **Chapter 5** articulates a pragmatic approach combining the different approaches discussed in Chapter 4.
>
> **Chapter 6** proposes a formalism based on the pragmatic approach presented in Chapter 5. The purpose of this formalism is to represent the dynamics of agent communication, analyze conversations, and help agents to participate in conversations in a flexible way.
>
> **Chapter 7** defines the semantics related to the pragmatic approach as a logical model. This chapter establishes the link between the pragmatics and the semantics of our agent communication proposal.
>
> **Chapter 8** proposes a verification method for dialogue game protocols. These protocols are specified using the unified framework presented in Chapter 7. In this chapter, a tableau proof system for the logical model specified in Chapter 7 is defined. This proof system is used in the verification method.
>
> **Chapter 9** presents a persuasion dialogue game protocol based on our approach. This chapter discusses the formal specification, implementation, and complexity analysis of this protocol.

We conclude this thesis by summarizing our contributions and identifying directions for future work.

Chapter 2

Agent Communication

In this chapter, we present and discuss some proposals in the domain of agent communication. We briefly present three main languages developed in this domain. We discuss their semantics and philosophical foundations. Finally, we highlight their limitations which we address in detail in the next chapters.

2.1 Introduction

An interesting characteristic of multi-agent systems is the principle that agents can function more effectively in groups. Agents are designed to autonomously collaborate with each other in order to satisfy both their internal goals and the shared external demands generated by virtue of their participation in agent societies. This type of collaboration depends on a sophisticated system of inter-agent communication. The language used by agents for this communication is the Agent Communication Language (ACL). The main objective of an ACL is to model a suitable framework that allows heterogeneous agents to interact and to communicate with meaningful statements that convey information about their environment or knowledge (Kone, 2000).

Over the last decade, two main ACLs have been proposed: the Knowledge Query and Manipulation Language (KQML) (Finin et al., 1995) and the Foundation for Intelligent Physical Agents' Agent Communication Language (FIPA-ACL). FIPA-ACL is in turn based on the ARTIMIS Communication Language (ARCOL). The formal specifications and the semantics of these languages are based upon the philosophical foundation provided by *Speech Act Theory*. The purpose of this chapter is to introduce these languages and their philosophical foundations.

The rest of this chapter is organized as follows. In Sections 2.2, 2.3 and 2.4, we present KQML, ARCOL and FIPA-ACL respectively. In Section 2.5, we discuss a taxonomy of ACL semantics. Section 2.6 introduces the notion of conversation protocols. The philosophical foundations of ACLs are explained in Section 2.7. The final discussion (Section 2.8) evaluates the limits of these ACLs and establishes the link with the next chapters of this dissertation.

2.2 KQML

KQML arose from work sponsored by the American Government's Defense Advanced Research Projects Agency (DARPA). It is the result of research done by the Knowledge Sharing Effort (KSE), an initiative that aims at developing a foundation for software systems interaction and interoperability. Three working groups compose this consortium: the interlingua group, the shared and reusable knowledge base group, and the external interface group. The first group designed the Knowledge Interchange Format (KIF) as a common language for describing a message content. This format is an extension of first-order logic. The second group worked on the content of sharable knowledge bases. This group examined the problem of sharing the content of formally represented knowledge. The approach focused on common ontologies. Every knowledge-based system relies on some conceptualization of the world (objects, qualities, distinctions and relationships that matter when performing some task) that is embodied in concepts, distinctions, etc. using a formal representation. The group worked on the construction of ontologies for various domains. Each ontology, written in KIF, defines a set of classes, functions, and objects for some domain of discourse, and includes an axiomatization to constrain the interpretation. The third group produced the KQML language and looked at interactions of system components.

The language's primitives are called *performatives*. As the term suggests, the concept is related to *speech act theory* (Austin, 1962). Performatives define the permissible actions (operations) that agents may attempt when communicating with each other. KQML consists of a set of communication primitives aiming to support interaction between agents. In this language, an agent's mental attitudes (belief, intention, and desire) are expressed in the label of a message that represents a communicative act. A KQML message is conceptually divided into three levels (Labrou et al., 1999): (1) the communicative level which specifies the sender and receiver agents; (2) the message level which mainly specifies the type of performatives (affirmation, question, etc.), the language (KIF, Prolog, etc.) and the used ontology; (3) the content level, which specifies the message content. An example of KQML message is the following:

```
(tell
        :sender X
        :receiver Y
        :in-replay-to id1234
        :ontology Software
        :language Prolog
        :content (Price MathType 150)
)
```

The goal of the tell KQML performative is to convey to some receiving agent that the sending agent believes that a particular proposition (contained in the content field) is true. The example indicates that agent *X* answers a message of agent *Y* about the price of a software. It uses the Prolog language to describe the content and a particular ontology (Software) which indicates the significance of "MathType" and the currency associated with the value "150".

Initially, KQML was proposed without a defined semantics. This criticism led researchers to define a new language: the FIPA-ACL. The early version of KQML presented some confusions and ambiguities in the usage of the performatives. Later on, its authors gave it a semantics and limited the use of some performatives in order to avoid some of these problems. The new semantics is defined in terms of: 1) preconditions on the mental states of the sender and the receiver before the communication of the message, 2) postconditions that should hold after the message is sent and 3) completion conditions that indicate when the perlocutionary effect has been fulfilled. However, this semantics provides no semantic model for mental attitudes.

2.3 ARCOL

The ARTIMIS agent technology developed by France Telecom is a generic framework for instantiating communicating agents. This technology is based on the proposal of Sadek et al. (1997). In ARTIMIS, an agent can cooperatively interact with humans as well as with other agents. Agents' communicative acts are modeled as rational actions. Agents can reason about knowledge and actions pertaining to their communicative acts. ARCOL (ARTIMIS Communication Language) is the ACL used in ARTIMIS. An ARCOL expression relies on a semantic language SL for the definition of its semantics. SL, in turn, uses the language SCL (Semantic Content Language) to describe the semantics content of a communicative act. ARCOL includes the following set of primitives:

Inform: An agent uses the assertive act Inform to convey a message to another agent provided that it believes the content of this message.

Request: The directive act request enables an agent to demand an action from another agent provided that it has the capabilities to perform that action.

Confirmation: When the sender believes that the receiver is uncertain about the information being transmitted, this communicative act can be used to confirm it.

Inform referent: This communicative act enables an agent to inform another agent of the value of a referent with a given description.

The most important characteristic of the ARCOL language is its formal semantics as a reliable support for interoperability. However, ARCOL's fixed context with the sender agent required to be sincere is an impediment to heterogeneity.

2.4 FIPA-ACL

FIPA-ACL arose from attempts to develop an industry standard for agent communication. Its formal model and semantic language draw from the ARCOL Language. Conceptually, FIPA-ACL distinguishes two levels in communication messages. At the inner level, the content of messages can be expressed in any logical language. The outer level describes the locutions that agents can use in their communication. The content of messages is wrapped in these locutions. FIPA-ACL specifies 22 locutions (FIPA-ACL, 2001b). Here we mention some of them:

Accept Proposal: The action of accepting a previously submitted proposal to perform an action.

Agree: The action of agreeing to perform some action, possibly in the future.

Call for Proposal: The action of calling for proposals to perform a given action.

Confirm: The sender informs the receiver that a given proposition is true, where the receiver is known to be uncertain about the proposition.

Inform: The sender informs the receiver that a given proposition is true.

Not Understood: The sender informs the receiver that it received a message that it does not recognize or it is unable to process the content of this message.

Propose: The action of submitting a proposal to perform a certain action, given certain preconditions.

Query If: The action of asking another agent whether or not a given proposition is true.

Request: The sender requests the receiver to perform some action or a communicative act.

FIPA-ACL is an agent communication language whose developpement involved several parties in industry and academia. It lays out the practical components of agent communication and cooperation and a well-defined formal semantics. However, some practical applications pointed out several limitations of the FIPA standard (Kone, 2000). For example, this standard provides no support for real-time and performance requirements of telecommunication applications. In addition, FIPA-ACL semantics rests only on the belief states of communicative agents. In this context, the sender does not guarantee the actual accomplishment of the expected outcome at the destination because the semantics offers no mechanism on how to infer the mental state of the receiving end.

2.5 A Taxonomy of ACL Semantics

As stated by McBurney (2002), when considering formal languages, different semantics can be defined viewing them as a mathematical logic. A semantics is a relationship between the language and a space M of mathematical structures, called *models*. A statement S in the language specifies a subset $M(S)$ of M. Such a statement is said to be true in a particular model M_0 if $M_0 \in M(S)$. A statement is said to be logically valid if it is true in every model, i.e., if $M(S) = M$.

Another type of semantics is derived from linguistics. As expressed by the linguist Morris (1938), the syntax of a language is *the formal relation of signs to one another* and the semantics of the language defines *the relations of signs to the objects to which the signs are applicable*. Thus, it makes sense to speak of the truth of a sign, since this indicates that the sign has a relationship to external objects in the world (McBurney, 2002).

Because ACLs are formal languages, their semantics can be defined as in mathematical logic. However, because they are also intended as communication mechanisms, a linguistics-based semantics can be defined. In this section, we are only interested by formal semantics defined from mathematical logic. We can distinguish five formal semantics: *axiomatic* semantics, *operational* semantics, *denotational* semantics, *game-theoretic* semantics, and *tableau* semantics[1].

Axiomatic semantics is defined by a set of assertions about properties of a system and how they are affected by program execution. For ACLs, this semantics defines each locution in terms of the preconditions which must be fulfilled before the locution can be uttered, and in terms of the post-conditions which must become true after the production of the utterance. We distinguish between *public* and *private* axiomatic semantics. In public axiomatic semantics, the pre-conditions and post-conditions describe states or conditions of the dialogue which can be observed by all participants. In private axiomatic semantics, pre and post-conditions describe states or conditions which are internal to one or more of the participants and thus are not directly observable by the others. For example, the semantics defined for FIPA-ACL and KQML are private axiomatic semantics. For example, the *Inform* FIPA-ACL act, in which one agent tells another some proposition, may only be uttered if the first agent believes the proposition to be true. This is termed a *sincerity* condition. This semantic approach, based on mental notions such as beliefs and intentions, will be detailed in Chapter 4. On the other hand, the semantics provided for argument-based ACLs is a public axiomatic semantics (Amgoud et al., 2002). For example, according to this semantics, an agent *a* which asserts a proposition *p* is supposed to have an argument in favor of it. Thus, if this proposition is attacked by another agent, agent *a* must defend it.

Operational semantics is defined by a set of rules specifying how the state of an abstract machine changes while executing a set of instructions. Each rule specifies certain preconditions on the contents of states and their new contents after the application of the rule. In the context of ACLs, operational semantics considers the locutions as instructions which operate successively on the states of some abstract machine. This semantics defines the locutions in terms of the transitions they apply on the states of this machine. van Eijk and his colleagues (2000) studied operational semantics for ACLs on the basis of an agent communication programming language which is a formal framework that identifies basic aspects of agent communication. The formal semantics of this language is given by means of transition rules that describe its operational behavior. Moreover, the operational semantics closely follows the syntactic structure of the language, and hence gives rise to an abstract machine to interpret the language.

Denotational semantics is a technique for describing the meaning of programs in terms of mathematical functions. Programs are translated into functions whose properties can be

---

[1] McBurney (2002) only distinguished four formal semantics: *axiomatic* semantics, *operational* semantics, *denotational* semantics, and *game-theoretic* semantics.

proved. In denotational semantics, each element of the language syntax attributes a relationship to an abstract mathematical entity, its denotation. The possible world semantics related to modal languages is an example of such semantics (Hintikka, 1962) (Kripke, 1963). For example, the semantics of the *necessarily* and *possibly* modal connectives are given by introducing an *accessibility relation* into models for the language. This relation defines what worlds are considered accessible from every other world. A formula is necessarily true if it is true in every world accessible from the current world, and it is possibly true if it is true in at least one world accessible from the current world. An example of this semantics is given in Chapter 7 of this dissertation. Another example is given by Parsons (1997). In this semantics, argumentation systems are connected to qualitative probabilistic networks. Propositions correspond to nodes in these networks and arguments between propositions correspond to the associated nodes. In order to use the denotational semantics approach, we must be able to derive the semantic meaning of a statement expressed in the language from the semantic meaning of its elements, this property is called compositionality.

In game-theoretic semantics, each well formed formula in a language is associated with a formal game between two players: a protagonist and an antagonist. A statement is considered to be true when and only when a winning strategy exists for the protagonist in the associated game. McBurney (2002) proposed a game-theoretic semantics for an inquiry dialogue protocol. In this semantics, a winning strategy for a player is a set of rules enabling the player to move in such a way that executing these moves guarantees that the player can win the game, no matter which moves are made by the opposing player.
Tableau semantics is based on the use of assertions and proof rules. The proof rules are inference rules aiming to prove the truth or falsity of the assertions. Unlike traditional proof systems which are bottom-up approaches, tableau semantics uses a top-down or goal-oriented approach. Proof rules are used in order to prove a certain formula by inferring when a state in a Kripke structure satisfies such a formula. According to this semantics, we start from a goal, and we apply a proof rule and determine the sub-goals to be proven. The proof rules are designed so that the goal is true if all the sub-goals are true. In ACLs, this semantics can be used to give the meaning of communicative acts by considering them as goals, and then determining the sub-goals by applying a set of proof-rules. To our knowledge, the only tableau semantics defined in the context of agent communication is the one that we propose in Chapter 8 of this dissertation.

2.6 Conversation Protocols

When conversing, agents do not exchange isolated messages, but a sequence of interdependent messages. To take into account this aspect, FIPA proposes to use conversation protocols (also called conversation policies). Conversation protocols are general constraints on the sequences of semantically coherent messages leading to a goal (Greaves et al., 2000). The coherence of messages is ensured by these constraints. These protocols are specified as static structures which define in a deterministic way the order in which communicative acts are connected. Like protocols used in distributed systems, these structures are generally modeled using finite state machines (Winograd and Flores, 1986) (Barbuceanu and Fox, 1995), Petri nets (traditional or extended) (Cost et al., 2000) or UML sequence diagrams. The idea of protocols is to facilitate the task of computing the possible answers to a given message. *Request Interaction Protocol*, *Contract Net Interaction Protocol*, and *English Auction Interaction Protocol* (FIPA, 1997, 1999, 2001a) are examples of such protocols.

As outlined by Greaves and his colleagues (2000) and Vongkacem and Chaib-draa (2000), conversation protocols must address two fundamental issues:

Flexibility: The aim of conversation protocols is basically to constrain the conversational behavior of the participants while taking into account the fact that agents are autonomous. These protocols must find equilibrium between the normative aspect ensured by the constraints and the flexibility expected in multi-agent communications.

Specification: Conversation protocols must be specified while taking into account the computational complexity of reasoning about them. For example, this specification must avoid the state-explosion problem when analyzing a sequence of utterances in order to decide which locution to utter next. In addition, these protocols should be designed in such a way that a formal verification is possible. The verification of some properties in these protocols, for example, deadlock, termination, correctness, etc. is extremely important in open environments.

2.7 From the Philosophy of Language to Agent Communication

2.7.1 From Speech Act Theory to Conversations

The specifications of KQML, ARCOL, and FIPA-ACL are based on a philosophical theory called *speech act theory*. This theory is due originally to a philosopher of language, Austin (1962), and extended by Searle (1969, 1983) and Searle and Vanderveken (1985). It considers human utterances as actions, in that they may change the state of the world. Speech is not just used to designate something, it actually does something. This explains the use of the word "act" in the description of ARCOL and FIPA-ACL locutions. According to Searle, to understand language, one must understand the speaker's intention. Since language is intentional behavior, it should be treated like a form of action. Thus, Searle refers to utterances as speech acts. The speech act is the basic unit of language used to express meaning, an utterance that expresses an intention. In general, speech acts are acts of communication. To communicate is to express a certain attitude, and the type of speech act being performed corresponds to the type of attitude being expressed. For example, a statement expresses a belief, a request expresses a desire, and an apology expresses a regret. As a communicative act, a speech act succeeds if the audience identifies, in accordance with the speaker's intention, the attitude being expressed.

Speech act theory identifies three distinct levels of action beyond the act of utterance itself. It distinguishes the act *of* saying something (the "*locutionary*" act), what one does *in* saying it (the "*illocutionary*" act), and what one does *by* saying it (the "*perlocutionary*" act). Speech acts, being perlocutionary as well as illocutionary, generally have some ulterior purpose, but they are distinguished primarily by their illocutionary type, such as *asserting*, *requesting*, *promising* and *apologizing*, which in turn are distinguished by the type of attitude expressed. The perlocutionary act is a matter of trying to get the hearer to form some correlative attitude and in some cases to act in a certain way. For example, a statement expresses a belief and normally has the further purpose of getting the addressee form the same belief. A request expresses a desire for the addressee to do a certain thing and normally aims for the addressee to intend to and, indeed, actually do that thing. A promise expresses the speaker's firm intention to do something, together with the belief that by his utterance he is obligated to do it, and normally aims further for the addressee to expect, and to feel entitled to expect, the speaker to do it.

As outlined by Vanderveken (2001), speech act theory tends to study isolated illocutionary acts performed by using sentences in single context of utterance. However, it is clear that speech acts are seldom performed alone. Speakers perform their illocutionary acts within entire conversations in order to achieve common goals such as discussing news, coordinating their joint actions or negotiating. For this reason, Vanderveken proposed a *theory of discourse* enriching Speech Act Theory. The purpose of this theory is to analyze the structure of conversations whose type is provided with an internal discursive purpose and to provide a taxonomy of these conversations. This taxonomy is based on the fact that there are only four possible discursive goals that speakers can attempt to achieve by way of conversing: the *descriptive*, *deliberative*, *declarative*, and *expressive* goals. These goals correspond to one of the four possible directions of fit between words and things. Using these directions, the four conversation types can be described as follow:

1. Conversations with the *words-to-things direction of fit* have the descriptive goal: They serve to describe what is happening in the world. Such are descriptions, debates on a question, persuasions, arguments, explications, interrogations, etc.

2. Conversations with the *things-to-words direction of fit* have the deliberative goal: They serve to deliberate on which future actions speakers and hearers should commit themselves to in the world. Such are deliberations, negotiations, bargaining sessions, a compromise or the signing of a contract, auctions, etc.

3. Conversations with *the double direction of fit* have the declarative goal: They serve to transform the world by way of doing what one says. Such are official declarations like declarations of war or of independence, nominations, appointments, etc.

4. Conversations with *the empty direction of fit* have the expressive goal: They serve to express common attitudes of their speakers. Such are the exchanges of greetings, welcomes, congratulations, etc.

2.7.2 Walton and Krabbe's Classification

Another taxonomy of dialogues was proposed by two philosophers of argumentation, Walton and Krabbe (1995). In their book: *Commitment in Dialogue*, *Basic Concepts of Interpersonal Reasoning*, Walton and Krabbe distinguish six main types of dialogues:

1. Persuasion, which is centered around conflicting points of view.

2. Negotiation, in which participants aim to achieve a settlement that is particularly advantageous for individual parties.

3. Inquiry, in which the aim is to collectively discover more information, as well as to destroy incorrect information.

4. Deliberation, which is driven by the need to take a collective decision.

5. Information-seeking, in which one party asks for information known by another.

6. Eristic, in which two parties combat each other in a quarrel.

While Vanderveken's classification is based on directions of fit between words and things, Walton and Krabbe's classification is based upon two factors: the initial situation and the goal of the dialogue. Table 2.1 illustrates these factors.

Dialogue type	Initial situation	Dialogue goal
Persuasion	Conflicting point of view	Resolution of conflict
Negotiation	Conflict of interest	Making a deal
Inquiry	General ignorance	Growth of knowledge
Deliberation	Need for action	Reach a decision
Information-seeking	Personal ignorance	Spreading knowledge
Eristic	Antagonism	Accommodation in relationship

Table 2.1. Walton and Krabbe's classification

These six types may be refined into subtypes, simply by specifying more elaborate conditions on the dialogues (e.g. the type of conflict or the degree of rigidity of the rules). For example, a dispute is a subtype of persuasion, where each participant tries to defend its point of view. In addition, this taxonomy is based on an argumentation vision and it coincides with the *dialectical systems* proposed by Hamblin (1970). These systems will be discussed in detail in Chapter 4, Section 4.4.2.

Walton and Krabbe introduced the notion of *dialectical shift* to capture the change in the context of dialogue during a conversation from one type of dialogue to another. Indeed, dialogues are usually not of a single type from their beginning to their end. For instance, it is common to start an inquiry dialogue, to realize during the dialogue that there is a controversial issue at stake, to enter in a dispute sub-dialogue, and to eventually resume the inquiry dialogue when the issue has been resolved. This notion allows us to construct complex dialogues by combining different types.

Vanderveken's classification can be regarded as more general than Walton and Krabbe's one in the sense that the dialogue types discussed by Walton and Krabbe are subtypes of the four types proposed by Vanderveken. Persuasion, Inquiry, and Information-seeking are conversations with a descriptive goal, negotiation and deliberation are conversations with a deliberative goal, and Eristic is a conversation with an expressive goal.

2.8 Discussion

KQML, ARCOL, and FIPA-ACL have the advantage of being based on a theory largely studied by philosophers of language. They are also formally specified using a modal logic. However, their private axiomatic semantics does not provide any technique for checking the agents' compliance to this semantics. In other words, it is not possible to verify whether or not the agents' communicative behavior matches their mental states. To remedy this, the semantics must also take into account the public (the observable) attitudes and the argumentative considerations. Such considerations enable us to explain the reasons behind the performance of communicative acts and how an agent can decide about the next act to be performed. In Chapter 5, we show how a combined approach (mental, social and argumentative approach) can resolve this problem.

Although FIPA protocols are practically interesting and can successfully be used in simple applications, they are not flexible enough to be used by autonomous agents in flexible and complex conversations such as persuasions, argumentative negotiations, deliberations, etc. When the allowed communicative acts are limited and the purpose of the communication is just to exchange some messages, rigid protocols provide an interesting solution and there is no need to define supplementary mechanisms. However, when agents must participate in complex conversations, for example in order to persuade each other, to negotiate, to deliberate, etc., they should act autonomously in the sense that they must be able to make decisions and to take initiatives. In this case, rigid protocols are not suitable. This is due to the fact that agents must follow the whole protocol in order to communicate and it is not clear how agents can make choices between several possible actions. For this reason, it is preferable to use small conversation protocols that can be put together to construct complex protocols. The combination rules must be formally specified and agents must be able to reason about these protocols in order to be able to use them flexibly. In the next chapter, we discuss dialogue game protocols aiming to address this issue. In Chapter 6, we propose a framework enabling agents to reason about their communicative acts and in Chapter 9, we show how this framework can be used to specify a flexible dialogue game protocol.

On the other hand, KQML and FIPA-ACL grew from efforts by DARPA to develop technologies for *knowledge sharing* (Labrou et al., 1999). Such a conceptual paradigm explains why several communicative acts seek to request or send information (e.g. *Inform*, *Inform-if*, *Conform*, *Query*). Despite this, these languages have not been designed with the possibility that such information may be questioned or challenged. An agent receiving an *Inform*(φ) message who is unsure about the truth of its content φ, or who does not hold the belief that φ is true, has few options to express these views. In addition, these languages have been developed without thinking that an agent can justify or defend its beliefs, or seek to persuade another to change its beliefs. This is due to the absence of a logic of argumentation in the specification of these languages. Such a logic is extremely useful for capturing the agents' reasoning. In Chapters 3 and 4, we present some proposals using this type of logic. In Chapters 5 and 6, we present our proposal based on this logic and in Chapter 7, we present a modal semantics capturing this logic and its relation to the social approach.

Chapter 3

Dialogue Games

In this chapter, we go through some relevant proposals in dialogue game frameworks for agent communication. We highlight the foundations and the structures of these frameworks. We also compare these proposals and discuss their limits. Our dialogue game protocol presented in Chapter 9 is an attempt to push these limits.

3.1 Introduction

To communicate, agents using traditional agent communication protocols, like those proposed by FIPA, must follow the protocol sequences. Hence, these protocols are often unsuitable for autonomous agents. This is due to the inflexibility of these protocols and to the fact that there is no mechanism allowing agents to choose the communicative acts they will perform. To solve this problem, several proposals have been put forward using formal dialogue games. Formal dialogue games are abstract structures that can be composed to construct the whole dialogue. They involve interactions between two or more players, in which each player moves by making utterances according to a pre-defined set of rules. The rules typically define which locutions may or must be uttered in different circumstances, and they may also indicate when the dialogue terminates. As a joint activity, the dialogue requires the coordination of the participants' actions. In this context, dialogue games are structures enabling agents to coordinate the dialogical activity.

Dialogue games have been studied in philosophy from at least the time of Aristotle (350 B.C) (van Emeren et al., 1996), and were extensively studied and practiced in medieval times (Spade, 1979). They differ from the games of Economic Game Theory, in that payoffs for winning or losing a game are not considered, and because there is no use of uncertainty measures such as probabilities, to model the possible moves of opponents. Dialogue games have been used in argumentation theory for the contextual analysis of fallacious reasoning, on the assumption that what may count as a logical fallacy in one context may not be so in another. The main proponents of this approach were Hamblin (1970, 1971) and MacKenzie (1979, 1990). All Hamblin's games have as their purpose, "the exchange of information among the participants" and so may be considered as models of information-seeking dialogues.

Another strand of philosophy, led by Lorenzen (1960), has used formal dialogue games to provide a constructive proof-theory for statements in intuitionistic and classical logic. Here a speaker in a dialogue game treats a proposition in a logical language as a statement. This statement is subject to question and challenge by an opponent. The proponent of the statement must defend the statement against the opponent's attack in pre-defined ways. In doing so, a proof (or disproof) of the statement is incrementally constructed. The precise rules of the dialogue game determine whether this proof corresponds to classical or intuitionistic logic.

Recently, dialogue games have been proposed as a basis for agent communication. Various dialogue game protocols have been developed. Applications have included frameworks for Walton and Krabbe's analysis of dialogue types (Reed, 1998) (McBurney and Parsons, 2001, 2002), for negotiation protocols (Dastani et al., 2000), and for agent team-formation dialogues modeled as combination of information-seeking and persuasion dialogues (Dignum et al., 2000). Dialogue games have also been used for joint-intention-formation dialogues, modeled as persuasion dialogues possibly containing embedded negotiation dialogues (Dignum et al., 2001), for request for action (Maudet et al., 2002), and for inconsistent and biased information (Lebbink et al., 2004). In this chapter, we go through these proposals in some details.

The rest of this chapter is organized as follows. In Section 3.2, we summarize Reeds' dialogue frames. In Section 3.3, we present Dastani et al.s' negotiation protocols. In Section 3.4, we discuss the layer model of McBurney and Parsons. Maudet et al.'s DIAGAL language and Lebbink et al.s' dialogue games will be presented in Sections 3.5 and 3.6. In Section 3.7 we review Dignum et al.s' dialogue games. Finally, in Section 3.8, we compare and discuss these proposals.

3.2 Reeds' Dialogue Frames

Reed (1998) proposed the notion of *dialogue frames* as abstract exchange structures. This notion is used to explore the dialogue typology proposed in (Walton and Krabbe, 1995) and one of its important features, the concept of functional embedding. In these frames, persuasion, inquiry and information seeking are *epistemic*, negotiation is concerned with *contracts*, and deliberation with *plans*. Epistemic issues can be modeled by a BDI approach (Rao and Georgeff, 1991), or a propositional logic encompassing beliefs, values (such as those employed to evaluate issues during negotiation), rules, intentions, etc. The notion of contract is intended to abstract from the precise structure used to reach a deal. Plan refers to the abstract notion of a set of partially ordered contracts. The foundation of the model is a set of agents, *A*, a set of agent's beliefs, *B*, a set of agent's contracts, *C*, and a set of agent's plans, *P*.

Contracts are composed of <*issue*, *value*> pairs. To make explicit the assumption that there is some basic result of a fulfilled contract, this result can be expressed as a conjunction of beliefs. Let us consider the example of a contract specifying that for agent *a* to receive information from agent *b*, *a* must pay b \$10. The issue-value pairs are <*Price*, \$10>, <*Quality*, *High*>. Plans can be constructed from contracts: a complete plan is a fully ordered set of contracts each of which is fully specified with respect to its result, *r*, its list of issue-value pairs, *vn* , and the settings of both issue and value in each value pair *vi*.

The set of dialogue types D is defined on the basis of the sets defined above (first paragraph of this section). Each type is a name-substrate pair:

$$D = \{ < persuade, B >, < negotiate, C >, < inquire, B >, < deliberate, P >, < infoseek, B > \}$$

Formally, a dialogue frame is defined as a tuple with four elements:

$$F = << t, \Delta > \in D, \tau \in \Delta, \left\{ u^0_{x_0 \to y_0, \dots,} u^n_{x_n \to y_n} \right\} >$$

where t is the type of this dialogue frame, Δ is the set of beliefs, contracts or plans, τ is the topic of the dialogue frame, $x_0, y_0 \in A$ are the interlocutors, and $u^i_{x_i \to y_i}$ refers to the ith utterance occurring in a dialogue between agents x_i and y_i, in which x_i is the originator of the utterance $x_i = y_{i+1}$ and $y_i = x_{i+1}$. An utterance $u^i_{x_i \to y_i}$ is a pair $< s, \left\{ \sigma_{0,\dots,} \sigma_n \right\} >$, in which s is a statement (i.e. a well formed formula in the communication language), and the $\sigma_i \in B$ represent the arguments supporting that statement.

A dialogue frame is thus of a particular type, t, and focuses on a particular topic. For example, for a persuasion dialogue, the topic focuses on a particular belief, and for a negotiation dialogue, the topic focuses on a contract. A dialogue frame is initiated by a *propose-accept* sequence that can be considered as meta-acts whose purpose is to open the frame. These meta-acts have an empty support $\{\}$. The frame terminates with a characteristic utterance indicating acceptance or concession to the topic on the part of one of the agents.

Let us consider the following example of persuasion dialogue between two agents *a* and *b*.

$u^0_{a \to b} : << propose(persuade, has(c, information)) >, \{\} >$
$u^1_{b \to a} : << accept(persuade, has(c, information)) >, \{\} >$
$u^2_{a \to b} : << tell(has(c, information)) >, \left\{ told\_by(has(c, information), d) \right\} >$
$u^3_{b \to a} : << tell(unreliable(d)) >, \{\} >$
$u^4_{a \to b} : << concede(unknown(has(c, information))) >, \{\} >$

In this example, agent *a* initiates the dialogue to persuade agent *b* that some third party, *c*, has information. The dialogue is open because agent *b* accepts it. Agent *a* supports its claim by citing *d* as its source. Agent *b* undercuts the argument by pointing out the unreliability of *d*, and with no further supports available, agent *a* retracts its assertion with a *concede* which terminates the dialogue frame.

Reed considers two kinds of game compositions, *sequencing* and *embedding*. *Sequencing* is the canonical ordering and *embedding* is captured within the model without further complications of the structures. Indeed, since propositions to enter a frame are moves like any others, they can be made within ongoing frames. When a new dialogue frame ϕ_1 is proposed at turn *i* by *a*, and accepted by *b* at *i*+1 while a frame ϕ_0 was open, Reed assumed that ϕ_0 is just suspended (ϕ_1 is then embedded in ϕ_0). When the frame terminates, ϕ_0 resumes where it was stopped. Generally, the speaker who concedes in the embedding frame is not the speaker who resumes in the embedding frame.

3.3 Protocols proposed by Dastani and his Colleagues

Dastani, Hulstijn and der Torre (Dastani et al., 2000) proposed a methodology for constructing flexible negotiation protocols based on joint actions and dialogue games, following the work of (Hulstijn, 2000a, b). Negotiation is considered as a combination of joint actions represented by simple dialogue games from which larger interactions can be constructed. These dialogue games consist of initiatives followed by responses.

The key notion of these negotiation protocols is coherence. An utterance or move in a negotiation dialogue is coherent with the dialogue context, if (i) it fits a plan that might achieve the apparent goals of the agent, and (ii) it fits the current interaction rules. The information conveyed or requested by an utterance is called the *semantic content*. An utterance has a purpose: the *communicative function*. Each utterance is analyzed as a dialogue act which is characterized by a semantic content and a communicative function.

Negotiation dialogue games are sequences of moves. Each move corresponds to a type of utterance. Moves can be either initiatives or responses. Each initiative must be followed by an appropriate response, although there may be other exchanges first. For example, a clarification exchange may precede the answer to a question. The basic game structure is an exchange capturing that an initiative can be followed by either a positive or a negative response, or else a retry. For example, a proposal is an initiative, an acceptance is the corresponding positive response, a rejection is the negative response, and a counter-proposal is an example of a retry. An exchange is allowed, given that the coherence constraint on the semantic contents of the initiative and response is met. In other words, the response must address the initiative. Formally, an exchange between two agents, a and b about the content ζ (the response content) is specified as follows:

$$\begin{aligned} exchange(a,b,\zeta) = initiative(a,b,\eta);\ & pos\_response(b,a,\zeta) \\ & \mid neg\_response(b,a,\zeta) \\ & \mid retry(b,a,\xi) \end{aligned}$$
$$\text{where } M_{a,b} \vDash coherent(\eta,\zeta)$$

$M_{a,b}$ is the shared dialogue context, η is the initiative content, and ξ is the retry content.

Games can be composed by sequencing or chaining. A sequential combination is specified as follows:

$$game(a,b,(\eta.\zeta)) = exchange(a,b,\eta); game(b,a,\zeta)$$
$$\text{where } M_{a,b} \vDash coherent(\eta,\zeta)$$

The recursive nature of the definition indicates that it is possible to combine as many games as requested. Like in the basic exchange, some coherence constraints are stated between the games' topic. For instance, to be combined, games have to share a common subject matter. With regard to chaining composition, constraints require the last dialogue act (reactive) of the first game being the first (initiative) of the second game. Canonical examples of such chaining structures are question / answer / evaluation or proposal / counter-proposal. The difference between sequencing and chaining is that unlike chaining, sequencing does not impose any constraint about the relationship between the games.

3.4 The Layer Model of McBurney and Parsons

McBurney and Parsons defined a model for a generic dialogue game protocol to represent combinations of dialogues according to the typology proposed by Walton and Krabbe (1995). This model is used in the development of a three-level hierarchical formalism for agent dialogues. The lowest level is the topic layer, the next level is the dialogue layer, and the highest level is the control layer. The topic layer defines the matters which may be discussed in the dialogue. These matters refer to real-world objects or to states of affairs.

In the dialogue layer, different dialogue games are modeled as classical dialectical systems[2] with the following components: (i) *beginning rules*, (ii) *locution rules*, (iii) *combination rules*, (iv) *commitment rules*, and (v) *termination rules*. Beginning rules define the circumstances under which the dialogue starts. Locution rules indicate which utterances are permitted. Typically, legal locutions allow participants to assert propositions, to question or contest prior assertions. They also allow agents to justify the propositions that they have asserted which have been subsequently questioned or contested. Combination rules define the dialogue contexts under which particular locutions are permitted or not. For instance, it may not be permitted for a participant to assert a proposition p and subsequently to assert the proposition again in the same dialogue, without in the meanwhile having retracted the former assertion. Commitment rules define the circumstances under which participants express commitment to a proposition. These rules are inspired by formal dialogue systems proposed by Hamblin (1970) that establish public sets of commitments, called commitment stores, for each participant. This notion will be detailed in Chapter 4. Termination rules define the circumstances under which the dialogue ends.

The selection of specific dialogue types and transition between these types is presented in the control layer. This layer is defined in terms of two components: a set of *atomic dialogue types* which include the dialogue types of Walton and Krabbe, and a set of *control dialogues* which are dialogues that have as their discussion subjects other dialogues rather than topics. These dialogues include beginning and termination dialogues.

In addition, McBurney and Parsons propose the following combinations of atomic or control dialogues:

Iteration: If G is a dialogue, then G^n is also a dialogue consisting of the n-fold repetition of G. Each dialogue starts after termination of the preceding dialogue.

---

[2] The notion of the dialectical system will be discussed in Chapter 4, Section 4.4.2.

Sequencing: If G and H are both dialogues, then $G\ ;\ H$ is also a dialogue consisting of undertaking G until its closure and then immediately undertaking H.

Parallelization: If G and H are both dialogues, then $G \cap H$ is also a dialogue consisting of undertaking both G and H simultaneously until termination.

Embedding: If G and H are both dialogues, then $G[H|\Phi]$ is also a dialogue consisting of undertaking G until a sequence of legal locutions Φ of G has been executed, and then switching immediately to dialogue H which is undertaken until its termination, whereupon dialogue G resumes from where it was interrupted.

Testing: If p is a well formed formula, then $<p>$ is a control dialogue testing p truth status. When p is found to be false, the current active dialogue ends.

3.5 The DIAGAL Language proposed by Maudet and Chaib-draa

Maudet and Chaib-draa (2002) proposed an agent communication language DIAGAL (DIAlogue Game based Agent Language) by adapting the Maudet's work (2001) to the communication between software agents. An implementation of this language as a dialogue game simulator is described in (Labrie et al., 2003). In the model proposed by the authors, dialogue games are handled through a *contextualization game* which aims at defining how games are opened, combined, and closed during the conversation. This model adopts a strict commitment-based approach within the game structure. This approach proposed by (Singh, 1998) and (Colombetti, 2000) will be discussed in detail in Chapter 4.

In DIAGAL, games are bilateral structures capturing the different commitments created during the dialogue (Chaib-draa et al., 2005). These games are defined by *entry conditions*, *success conditions*, *exit conditions*, and *dialogue rules*. Entry conditions define conditions which must be fulfilled at the beginning of the game. Success conditions are conditions which indicate whether the game terminates successfully or not. Exit conditions define the goal of the participants when they are engaged in the game. Dialogue rules indicate the permitted communicative acts that participating agents can perform. In their formulation, the authors use sanctions penalizing agents that will not follow the expected dialogical behavior as described in the dialogue rules.

Maudet, Chaib-draa, and Labrie (Maudet et al., 2002) used DIAGAL to model the request for action proposed by Winograd and Flores (1986) as a composition of different basic games. These compositions which can have *conditions* and *effects* are:

Sequencing: denoted $g_1\ ;\ g_2$, which means that g_2 starts immediately after termination of g_1.
Conditions game g_1 is closed.
Effects termination of game g_1 involves entering g_2.

Choice: denoted $g_1 \mid g_2$, which means that participants play either g_1 or g_2 non-deterministically. This combination has no specific conditions nor consequences.

Pre-sequencing: denoted $g_2 \rightsquigarrow g_1$, which means that g_2 is opened while g_1 is proposed.
Conditions game g_1 is proposed.
Effects successful termination of game g_1 involves entering game g_2.
These pre-sequencing games are used to ensure that entry conditions of a forthcoming game are actually established.

Embedding: denoted $g_1 < g_2$, which means that g_1 is now opened while g_2 was already opened. This means that g_2 is suspended and one must return to it after the termination of g_1.
Conditions game g_1 is open.
Effects Commitments of the embedding games are considered proprietary over those of the embedding game.

Flores, Pasquier, and Chaib-draa (2004) proposed a conversational semantics for DIAGAL using social commitments. This semantics defines the meaning of messages on the basis of their use as coordinating devices advancing conversations. This semantics captures the evolution of conversations using the state of social commitments and the state of activities in which agents participate. According to the authors, a commitment could be either accepted or rejected according to whether or not agents are engaged in it. If accepted, a commitment is active, violated or fulfilled. If rejected, it is either inactive or cancelled. Commitments can move between states through four transition types: adoption, where an active commitment becomes accepted; violation and fulfillment, where an active commitment becomes violated or fulfilled, respectively; and discharge, where an accepted commitment becomes cancelled.

In this semantics, the meaning of communicative acts is defined through four levels: *compositional level*, *conversational level*, *commitment state level*, and *joint activity level*. Compositional level deals with message classification. Definitions at this level identify messages based on the type and identity of their components. Conversational level indicates the significance of messages once they are uttered. This significance is given taking into account the fact that messages as part of conversations seeking agreement to advance the state of commitments. Commitment state level refines the definitions of messages according to the shared state of the commitment being manipulated. Joint activity level refers to the meaning given to messages when they are used as part of joint activities. Definitions at this level are given in terms of the type of actions the commitments bring about, and in terms of the roles that interacting agents play in these actions.

3.6 Dialogue Games proposed by Lebbink and his Colleagues

Lebbink, Witteman and Meyer (Lebbink et al., 2004) proposed dialogue games in which coherent conversational sequences with inconsistent and *biased* beliefs are described at the speech act level. A belief is called "biased" when more evidence exists to believe than to disbelieve something or *vice versa*. In the former, the belief is said to be *biased true*, and in the latter, the belief is said to be *biased false*. A special case of a biased belief is when an agent has evidence to believe a statement but it also has an equal amount of evidence to believe the contrary. In such a situation, an agent's belief is considered inconsistent from an epistemic perspective.

The authors present these biased and inconsistent beliefs with *bilattice* structures (Fitting, 1991) that are constructed from two *complete lattices* $(B, \leq_1)$ and $(D, \leq_2)$. A complete lattice is a structure $(B, \leq)$ such that B is a non empty set ordered according to $\leq$ and for all $S \subseteq B$, there is a greatest lower bound and a least upper bound of *S*. A bilattice is an algebraic structure that formalizes an intuitive space of generalized truth-values with two lattice orderings B and D. The intuition is that B provides evidence for believing a statement and D provides evidence for disbelieving a statement. A bilattice has at least four truth-values: **t**, **f**, **u**, and **i**. Truth-value **t** represents full evidence for believing and no evidence for disbelieving. Opposite to **t** is truth-value **f** that represents no evidence for believing but maximal evidence for disbelieving. Truth-values **t** and **f** correspond to the *true* and *false* values of classical logic. In truth-value **u** neither evidence for believing nor for disbelieving exists. In truth value **i** both maximal evidence for believing and for disbelieving exist.

In addition, the authors define a multi-valued logic in order to describe dialogue games in which agents can communicate about their cognitive states. Whereas in classical logic terms are assigned a truth value *true* or *false*, in multi-valued logic, new truth-values can be captured to represent epistemic attitudes. These truth-values can represent unknown information and inconsistent and biased information. A language of multi-valued logic is defined in order to formalize two types of sentences: *atomic sentences* and *conditional sentences*. Atomic sentences consist of a propositional formula taken from an ontology o. Conditional sentences resemble the conditionals of classical logic.

In the dialogue games proposed by the authors, communicative acts are utterances used by agents to manifest parts of their cognitive states. Three communicative acts are used: *questions*, *statements of belief* and *statements of ignorance*. A question is a request for a belief addition, that is, an agent *a* asks an agent *b* whether it may add a sentence to its beliefs. In a statement of belief, an agent *a* states to an agent *b* that a given sentence is part of its beliefs and that *b* may add this to its beliefs. A belief statement can be an approval of a request for a belief addition. This request can also be denied, which is in effect a statement of ignorance, that is, an agent *a* states to an agent *b* that it is ignorant about a given sentence.

A dialogue game is formalized by, first, defining the agent's cognitive state as a set of multi-valued theories, second, by defining the dialogue rules, and last, by defining update rules. As a motivation to participate in a dialogue game, agents have the incentive to reduce an *imbalanced* desire and belief state (Grice, 1975). There is an imbalance in the agents' belief and desire state, when these agents do not believe a proposition but they do desire to believe it. In such case, it is said that the agent is interested to add the proposition to its beliefs. For example, two roles of questions are distinguished. The first role is to reduce the imbalance between an agent's desire and belief, that is, the question is about a sentence the agent itself is interested to believe. The second role is to reduce an imbalanced desire and belief state of another agent, that is, the question is about a sentence another agent is interested to believe. Dialogue rules define which communicative acts are applicable in a dialogue game. For example, a question from an agent *a* to an agent *b* is applicable when *a* is interested in a sentence and this sentence is *sensible*, that is, it is not part of *b*'s ignorance as *a* is aware of. In addition, the question must be *fresh*, that is, *a* is not allowed to pose a question for the same information more than once. Update rules prescribe the cognitive state of both the sending and the receiving agent after the information in a communicative act is accepted by both agents. For example, if an agent *a* has uttered a belief statement to an agent *b*, agent *b* believes the underlying sentence, *b* is aware that *a* believes the sentence, and *a* is aware that *b* believes the sentence. In fact, Dialogue rules and update rules describe pre and post conditions on agent's cognitive state.

3.7 Dialogue Games proposed by Dignum and his Colleagues

The dialogue game protocols presented in the work of Dignum, Duin-Keplicz and Verburgge (2000, 2001) are intended to enable agents to form teams and to agree on joint intentions. They present a theory for agents that are able to discuss the team formation and to adopt joint intentions and subsequently work as team members until the collective goal has been fulfilled. For both protocols, the authors assume that one agent, an *initiator* or *proponent*, seeks to persuade others (*opponents*) to join a team, and that another initiator (possibly the same agent) seeks, after the team formation, to persuade team members to adopt a group belief or intention. They present structured dialogues, with an emphasis on persuasion, which can be shown to lead to the required team formation and joint intentions. The dialogue games are formally specified using modal logics and speech acts. The team-formation dialogue is modeled as information-seeking dialogue followed by a persuasion, while the joint-intentions-formation dialogue is modeled as a persuasion dialogue, which may include embedded negotiation dialogues. For the persuasion dialogue, the authors adapt the rigorous persuasion dialogue game of Walton and Krabbe (1995).

The protocol for joint intention formation dialogues includes seven locutions: *statement*, *question*, *challenge*, *challenge with statement*, *question-with-statement* and *final remarks*. The statements associated with challenges and questions may be concessions made by the speaker. The protocol for team formation dialogues may also use the same set of locutions. The authors assume the participating agents have a belief-desire-intention architecture (BDI) and vest the locutions with a private axiomatic semantics, the locutions being defined in terms of their impacts on agent mental states.

For team formation by dialogue, the authors postulate that agent architecture should contain a number of specific modules. The heart of the system is the *reasoning* module. When realizing the consecutive stages leading ultimately to team formation, interaction with the *planning*, *communication* and *social reasoning* modules is necessary. All these modules contain a number of specific reasoning rules. Each rule refers to a specific aspect of the reasoning process.

The first task of the initiator in the team formation protocol is to form a partial plan for the achievement of the overall goal. For this reason, it determines which agents might be most suited to form the team. In order to determine this match, the initiator tries to find out the properties of the agents, being interested in three aspects, namely their *abilities*, *opportunities*, and their *willingness*. The initiator has to form beliefs about these aspects of the individual agents. Thus, it may first investigate the willingness of particular agents, and on this basis ask the interested ones about their abilities and opportunities. The questions in this stage form part of an information seeking dialogue game. To establish a collective intention within the team, agents start a persuasion dialogue consisting of three main stages: information exchange, rigorous persuasion, and completion.

3.8 Comparison and Discussion

In this section, we compare and discuss the dialogue game frameworks presented in this chapter using the following factors: the formal language used for the specification, the dialogue types supported by the framework, the architecture of the participating agents, the purpose of the proposal, the mechanism, if any, used in the framework for the decision making process, and the computational issues. Table 3.1summarizes this comparison.

Different logics are used to specify the dialogue game frameworks presented in this chapter. Modal and multi-valued logics are used to formalize and reason about agents' mental states. Nonmonotonic logic is used to formalize arguments that agents use to support their communicative acts. Other formal languages are also used to describe some elements such as the dialogue frames (Reed, 1998) and the contextualization game (Maudet and Chaib-draa, 2002). Lebbink et al. use a specific algebraic language to represent inconsistent information that is a part of the agent's cognitive state.

All the dialogue game proposals, with the exception of Maudet et al.'s framework and Lebbink et al.s' dialogue games, are based on the dialogue typology proposed by Walton and Krabbe. Reeds' dialogue frames and the layer model of McBurney and Parsons are defined to represent all the types according to this dialogue typology and the combination of these different types. Hence, these frameworks are more general than the other frameworks defined for specific dialogues. Maudet et al.'s DIAGAL is specified by four basic games: *Request* game, *Offer* game, *Inform* game, and *Ask* game. Although the combination of these games can describe different dialogue types, the authors do not specify these types. Lebbink et al.'s proposal does not use any philosophical foundation, but focuses on inconsistent dialogues without taking into account the goal of the dialogue.

Dastani et al. and McBurney and Parsons do not make assumptions concerning the internal architecture of agents. Consequently, it is not clear how these frameworks can be implemented and how agents establish the link between their mental states and their locutions during a dialogue. Reed does not specify a specific architecture, but only supposes that agents have epistemic issues, whose referent could equally be modeled by a BDI architecture or a propositional logic. On the other hand, Maudet and his colleagues propose an architecture in which each agent has a private agenda containing its commitments. Using this agenda, agents can follow the action effects on each move, i.e. check the creation, cancellation, fulfillment … of commitments. In addition, agents can use a shared action board representing the actions which were played during the dialogue. This board is represented as a history of the performed actions. In Lebbink et al.'s framework, agents have a cognitive state consisting of a set of mental constructs: beliefs, desires and ignorance. These constructs can be private or manifested (communicated explicitly). Dignum and his colleagues propose an architecture in which agents have beliefs, intentions and goals. In addition, agents can reason about these states and about other agents.

	Specification language	**Dialogue types**	**Agents' architecture**	**Purpose**	**Decision making process**	**Computation**
Reed	Private formal language + Nonmonotonic logic	Dialogue types of Walton and Krabbe	Abstract architecture	Modeling and analyzing dialogues	Partially supported	No computation
Dastani et al.	Modal logic	Negotiation + Information seeking	Unspecified	Constructing flexible negotiation protocols	Unsupported	No computation
McBurney and Parsons	Modal logic	Dialogue types of Walton and Krabbe	Unspecified	Representing combination of dialogues + generating dialogues	Supported	An operational semantics
Maudet et al.	Private formal language	Request + Offer + Information + Ask	Each agent has an agenda containing its commitments	Analyzing, modeling and verifying automated conversations	Unsupported	Implementation of a simulator
Lebbink et al.	Algebraic language + Multi-valued logic	Dialogues with inconsistent and biased information	Agents have a cognitive state	Analyzing inconsistent dialogues	Unsupported	No computation
Dignum et al.	Modal logic (KD_n45)	Persuasion + Information seeking + Negotiation	Agents have mental sates	Constructing agent dialogues for team formation	Partially supported	No computation

Table 3.1. Comparison of some dialogue game frameworks

Reed's dialogue frames do not specify the rules that govern the performance of communicative acts but only an abstract form of these acts. Consequently, the formalism is descriptive and not generative. The purpose of Reed's work is to analyze conversation, but cannot be used to help agents to take part in these conversations. Although the dialogue games proposed by Maudet et al. and by Lebbink et al. specify dialogue rules and update rules, these two formalisms do not specify how agents can generate dialogues. The reason is that they do not specify the decision making process enabling agents to decide, at a given moment, about the next communicative act to be performed. Dastani et al. propose a methodology to construct protocols by specifying some combination rules. However, because this methodology does not provide any decision making process, it does not specify how agents can use these protocols in an autonomy way. On the other hand, McBurney and Parsons's model and Dignum et al.s' dialogue games are defined for generating dialogues. The layer model is equipped with an argumentation theory that provides a decision making process and agents can reason about their locutions using dialogue layer rules. In the team formation dialogues, agents can also reason about their locutions. However, this reasoning mechanism is not clearly specified.

With the exception of the proposals of MacBurney and Parsons, and Maudet and his colleagues, there is no computational analysis in the other proposals. However, McBurney and Parsons propose only an operational semantics in order to achieve the objective of automating dialogues; they do not provide any implementation or complexity analysis. Operational semantics indicates how the states of a system change as a result of execution of the commands in a programming language. In dialogue games, the commands are the moves, and the states are the dialogue states which can be described by the different commitments. On the other hand, Maudet et al. provide a dialogue game simulator, but the computational complexity of the implemented dialogue games is not studied.

As a conclusion, the dialogue game frameworks discussed in this chapter have two main limitations. The first limitation is related to the link between private mental states, public or manifested states and the decision making process. This link is extremely important to generate dialogues and enable agents to participate flexibly in conversations. The second limitation is related to the computational issues. For example, complexity, termination and correctness of dialogue game algorithms should be analyzed when developing these algorithms. In addition, verifying whether agents respect or not these dialogue games protocols is another relevant issue to be addressed.

Chapter 4

A Taxonomy of the Proposed Approaches

In this chapter, we present our taxonomy of the proposed approaches in the domain of dialogue modeling and agent communication. We distinguish three main approaches: the mental approach, the social approach and the argumentative approach. The mental approach is based on the agents' private mental states like beliefs, desires, and intentions. The social approach highlights the importance of the public and social aspect of agent conversations. The argumentative approach uses the dialectical models discussed by the philosophers of argumentation.

4.1 Introduction

Communication between autonomous agents is widely recognized as a challenging research area in artificial intelligence and more particularly in the multi-agent systems community. Agent communication is at the intersection of several disciplines: philosophy of language, social psychology, artificial intelligence, logics, mathematics, etc. In a multi-agent system, agents may communicate in order to negotiate, to solve conflicts of interest, to cooperate, or simply to exchange information. All these communication requirements cannot be fulfilled by simply exchanging messages. Agents must be able to take part in coherent conversations which result from the performance of coordinated speech acts (Searle, 1969).

Over the years, important contributions have been made in modeling communication between software agents. Three main approaches have been proposed and applied to agent interactions and to agent communication languages (ACLs): the mental approach, the social approach, and the argumentative approach. Besides these approaches, some researchers proposed combined methods, called intentional-conventional approaches (Maudet, 2001). All these approaches originate from the research on the formalization of rational agents initiated by the pioneering work of Moore (1980) and Morgenstern (1986, 1987) in which knowledge and actions are considered.

In this chapter, we present and discuss these approaches on which our pragmatic approach presented in Chapter 5 is based. In Section 4.2, we present the mental approach. We summarize the model proposed by Cohen, Allen and Perrault, the rational interaction theory and other work. In Section 4.3, we present the social commitment approach. We discuss Singh et al.'s work, Colombetti et al.'s work and Flores and Kremer's work. In Section 4.4, we discuss the argumentative approach. We present the dialectical models and the use of argumentation for dialogue modeling. In Section 4.5, we briefly present some intentional-conventional approaches. In Section 4.6, we conclude the chapter by comparing the different approaches.

4.2 The Mental Approach

In the mental approach, so-called agent's mental structures (e.g. beliefs, desires and intentions: BDI) are used to model conversations and to define a formal semantics of speech acts. The objective of the BDI approach is to describe agents' rational behavior.

Beliefs are simply an agent's information at a given moment of time, i.e. what this agent believes to be true regarding the state of the world or other agents' knowledge. Desires represent the states of the world wished by an agent, without other consideration: it is completely possible to have unrealizable or contradictory desires. The process by which an agent selects, among these desires, those which could be pursued is deliberation. In order to select these desires, an agent can evaluate the feasibility of each desire. Other criteria like preferences between desires can also be considered (Hulstijn, 2000b). To define the concept of intention, many philosophical works have been put forward. For example, Bratman (1987) distinguishes doing something intentionally and intending to do something. Searle (1983) speaks about the intentions directed towards the future and the intentions in action. These two concepts are dependent since the intentions directed towards the future are generally related to the performance of intentional actions. The link between the concepts of goal and intention was discussed by many researchers. Some authors like Grosz and Kraus (1996) distinguish the notion of *intending that* (a proposition is performed), close to the notion of goal, and the notion of *intending to* (perform an action). The difference between these two concepts is that the first one does not necessarily involve an action performed by the agent itself.

In this section we summarize two main proposals in this approach: the plan-based models of Cohen, Perrault and Allen and the rational interaction theory of Cohen and Levesque.

4.2.1 Plan-based Models

Plan-based models of dialogue can be claimed to originate from three classic papers: Cohen and Perrault (1979), Perrault and Allen (1980), and Allen and Perrault (1980). These models admit the hypothesis that agents participating in a conversation have rational behaviors leading them to build and to execute plans in order to achieve some goals. The production of an utterance by a speaker is related to the performance of a communication sub-goal. The communicative actions are registered in the plans formulated by the conversational agents at the same level as the physical actions.

The notion of plan

Planning is the construction of a plan, from a model of the world, while respecting certain criteria. A plan is an organized set of actions whose performance enables agents to achieve a goal. A plan allows agents to anticipate a succession of actions in order to achieve this goal, i.e. a certain final state of the world. To introduce this notion, we consider the following example in which an agent *A* asks another agent *B* a question, to which the latter then responds. The presentation is taken from (Allen and Perrault, 1980):

A has a goal to acquire certain information. This causes him to create a plan that involves asking *B* a question. *B* will hopefully possess the sought information and answer the question. *A* then executes the plan, and thereby asks *B* the question. *B* receives the question and attempts to infer *A's* plan. In the plan, there might be goals that *A* cannot achieve without assistance. *B* can accept some of these goals as his own goals and create a plan to achieve them. *B* then executes its plan and thereby responds to *A's* question.

Plan inference is the process through which an agent *A* attempts to infer another agent *B's* plan, based on observed actions performed by *B*. Usually, this process starts with an incomplete plan, containing only a single observed action or an expected goal.

These two activities are modeled using the agents' cognitive components. To establish or recognize a plan, knowledge about the state of the world is needed in order to be able to modify this world and to reach the final state corresponding to the fixed goal. Agents also need to have knowledge about the means of achieving this goal. The participants also have beliefs about the world and knowledge and beliefs on the other participants. They finally have intentions to do an action and intentions to be in a certain situation.

Mental attitudes are omnipresent in plan-based models. The formalization of such attitudes is inspired by Hintikka's work (Hintikka, 1963). Allen and Perrault developed a modal logic in which the concepts of beliefs and knowledge are represented by the modal operators *BEL* and *KNOW*. This epistemic logic allows an agent to reason about what it knows and to deal with information that can be contradictory with its knowledge. There is no logical relation between what an agent *A* believes about another agent *B*'s beliefs and agent *A*'s own beliefs. For example, it is possible that agent *A* believes that a proposition *p* is true and believes that the agent *B* does not believe that *p* is true.

This epistemic logic is formalized as follows:
The formula $BEL(A, p)$ is read: "agent *A* believes that the proposition *p* is true". In modal logic and according to the semantics of possible worlds, this means that: if there is a world *M* in which the proposition $BEL(A, p)$ is true, *p* is true in all the accessible worlds from the world *M* by agent *A* using a belief accessibility relation. Worlds can be considered as a discrete sequence of events stretching infinitely into future (Cohen and Levesque, 1990). They can also be viewed as Kripke structures for a CTL-like logic (Rao and Georgeff, 1995) (Wooldridge, 2000). Intuitively, accessible worlds using a belief accessibility relation are the worlds that the agent believes possible. The formula $KNOW(A, p)$ is true if $BEL(A, p)$ is true and if *p* is indeed true. The authors assumed that the *BEL* operator satisfies the following axioms:

- $BEL(A, p) \Rightarrow BEL(A, BEL\ (A, p))$: transitivity
- $BEL(A, p) \Rightarrow \neg(BEL(A, \neg p)$: coherence
- $BEL(A, p) \wedge BEL(A, q) \Rightarrow (BEL(A, p \wedge q)$: conjunction
- $BEL(A, p) \vee BEL(A, p) \Rightarrow BEL(A, p \vee q)$: disjunction

- $BEL(A, p) \wedge BEL(A, p \rightarrow q) \Rightarrow BEL(A, q)$: rationality

To formalize speech acts as actions, the authors use the concept of *action schema*. An action schema is a rule described by a name, a set of parameters, and some formulae which are its *pre-conditions*, *effects*, and *body*. Preconditions are conditions that must be true if the action's execution is to succeed. Effects are conditions that become true after the action is executed. The body is a set of partially ordered goal states that must be achieved after performing the action. An action is intentional when its author wants to perform it. A speech act is an intentional action. The pre-conditions of such an action contain the formula *WANT*(*A*, *Action*). Figure 4.1 explains these notions for the *INFORM* speech act. The definition of INFORM is based on Grice's idea (Grice, 1957) that the speaker informs the hearer of something merely by causing the hearer to believe that the speaker wants him to know something. This is like an operation in planning.

***INFORM* (*A*, *B*, *p*)**
- Pre-conditions:
  - $WANT(A, INFORM\ (A, B, p)$
  - $KNOW(A, p)$
- Effect: $KNOW(B, p)$
- Body: $BEL(B,\ WANT(A, KNOW(B, p)))$

Figure 4.1. The action schema of INFORM speech act

Allen and Perrault identified three types of inference rules: the ones concerning actions, the ones concerning knowledge, and the ones concerning planning by others. Rules concerning actions are rules that support plan recognition. Four inference rules concerning actions are defined as follows:

*Precondition-Action Rul*e: If *P* is a precondition of an action *AC*T, and an agent *S* believes that another agent *A* wants to achieve *P*, then we can probably infer that *S* believes *A* wants *ACT* to be performed.

*Body-Action Rul*e: If *B* is part of the body of *AC*T, and if *S* believes that *A* wants *B* to be performed, it is likely that *S* believes that *A* may want to perform *AC*T.

*Action-Effect Rul*e: If *E* is an effect of an action *AC*T, and *S* believes *A* wants to perform *AC*T, then it is plausible that *S* believes that *A* wants the effect of that action.

*Want-Action Rul*e: If *S* believes that *A* wants another agent *N* to want some action *ACT* to be performed, then *S* may believe that *A* wants *ACT* to be performed.

Rules concerning knowledge define relations between goals of acquiring knowledge and goals and actions that use that knowledge. Rules concerning planning by others are construction rules that can be seen as the inverse of plan inference rules. The plan construction rules are: *Action-Precondition Rule*, *Action-Body Rule*, *Effect-Action Rule*, *Know Rule*, *Nested-Planning Rule*, and *Recognizing Nested-Planning Rule*. These rules resemblance to previously mentioned rules.

Several researchers explored the idea of using plans to model agent interactions and suggested different types of plans: domain plans and discourse plans (Litman and Allen, 1990), individual plans (Pollack, 1990), and shared plans (Grosz and Sidner, 1990). However, the fact that interaction is a dynamic activity and is dependent on the action context makes it difficult to model it using a planning approach. In particular, the plan recognition that is necessary to deduce other agents' intentions is extremely complex.

4.2.2 Rational Interaction Theory

Cohen and Levesque (1990) proposed an action theory upon which a rational interaction theory has been built. This theory is based on a modal logic whose semantics is given in terms of possible worlds. Action representation is based on dynamic logic. The corresponding language contains the usual connectives of a first-order language, operators for the propositional attitudes, as well as action expressions. These elements are:

$(BEL\ A\ p)$, $(GOAL\ A\ p)$: p follows from A's beliefs or goals.
$(BMB\ A\ B\ p)$: A believes that p is a mutual belief with B.
$(AGT\ A\ a)$: A is the only agent of action a.
$a \leq b$: action a is an initial subsequence of b. Action variables range over sequences of primitive actions.
$(HAPPENS\ a)$, $(DONE\ a)$: action a will happen next, action a has just happened.
$a\ ;\ b$: action sequence.
$a \mid b$ nondeterministic choice.
$p?$ test action.
a^* repetition.
$p?\ ;\ a$ action a occurring when p holds.
$a\ ;\ p?$ action a occurs after which p holds.

From these elements, the following abbreviations can be adopted:

$\Diamond p =_{def} \exists a\ (HAPPENS\ a\ ;\ p?)$
$(LATER\ p) =_{def} \neg p \wedge \Diamond p$
$\Box p =_{def} \neg\Diamond\neg p$
$(PRIOR\ p\ q) =_{def} \forall c\ (HAPPENS\ c\ ;\ q?) \supset \exists a\ (a \leq c) \wedge (HAPPENS\ a\ ;\ p?)$
$(KNOW\ A\ p) =_{def} p \wedge (BEL\ A\ p)$

To define the notion of intention, the authors use the notion of *Persistent Goal P-GOAL* that is an internal and individual commitment of agent. Formally:

$(P\text{-}GOAL\ A\ p\ q) =_{def}$
1- $(BEL\ A\ \neg p) \wedge$
2- $(GOAL\ A\ (LATER\ p)) \wedge$
3- $[KNOW\ A\ (PRIOR\ [(BEL\ A\ p) \vee (BEL\ A\ \Box\neg p) \vee (BEL\ A\ \neg q)]$

$\neg$[*GOAL A* (*LATER p*)])].

This definition indicates that the agent *A* believes that *p* is currently false, chooses that it will be true later, and knows that before abandoning this choice, it must either believe it is true, believe it never will be true, or believe that *q*, an escape clause (used to model sub-goals, reasons, etc.) is false.

In this theory, intention to do an action *a* is a kind of persistent goal in which an agent commits to do an action, in a particular mental sate. Formally:

(*INTEND A a q*) $=_{def}$ (*P-GOAL A* [*DONE A* (*BEL A* (*HAPPENS a*))? ; *a*]*q*).

A fundamental notion in Cohen and Levesque's theory is an *ATTEMPT*. This notion discussed by Searle (1969) is used to define the illocutionary acts. An attempt to achieve ψ via Φ by performing an action *a* is defined as follows:

{*ATTEMPT A a* ψ Φ} $=_{def}$ [(*GOAL A* (*LATER* ψ))
$\wedge$ (*INTEND A a* ; Φ? (*GOAL A* (*LATER* ψ)))]? ; *a*

This definition indicates that, before performing *a*, the agent *A* chooses that ψ should eventually become true, and intends that *a* should produce Φ relative to that choice. So, ψ represents some ultimate goal that may or may not be achieved by the attempt, while Φ represents what it takes to make an honest effort. Using this notion, the authors defined the semantics of some illocutionary acts. Figure 4.2 illustrates the case of the *INFORM* act.

{*INFORM A B a p*} $=_{def}$
{*ATTEMPT A a*
(*KNOW B p*)
[*BMB B A*
(*P-GOAL A*
(*KNOW B* (*KNOW A p*)))]}

Figure 4.2. Definition of INFORM in Cohen and Levesque's theory

The illocutionary act of informing is defined as an attempt by which the speaker (agent *A*) is committed (in the sense of persistent goal) to the addressee's knowing that *A* knows *p*. In other words, agent *A* is committed to the addressee's knowing in which mental state *A* is. Although *A* is committed to getting the addressee to believe something about its goals, what *A* hopes to achieve is for the addressee to come to know *p*. To achieve this goal, it is necessary that the addressee *B* shares with *A* the mutual belief that *B* knows that *A* knows that *p* is true.

The fundamental idea of this approach is that illocutionary acts can only be derived from the analysis of the agents' mental states. In addition, in Cohen and Levesque's framework an agent intends to do an action if it has the persistent goal to have done the action. This reduction of intentions to do actions for goals is criticized by (Meyer et al., 1999): although intentions to do actions should be related to goals, this relation should express that doing the action helps in bringing about some goal and not that doing the action in itself is a goal.

According to the rational interaction theory, cooperation and sincerity are the two characteristics on which the agents' rational behavior rests. Cooperation can take the form of very strict constraints, like the adoption of goals. An agent is cooperative when it adopts the goal of its addressee. Thus, recognizing the speaker's underlying goals, as precisely as possible, is necessary to offer cooperative answers to it. In addition, the semantics of speech acts is conditioned by the fact that the speaker is sincere and that the addressee believes that the speaker is sincere. For example, in the INFORM act, the speaker is assumed to be sincere when it is committed to the addressee's knowing its mental state.

4.2.3 Other Work

Shapiro, Lespérance and Levesque (1998) proposed a language for specifying and verifying communicating multi-agent systems called Cognitive Agent Specification Language (CASL). Extended by Shapiro and Lespérance (2001) and Shapiro et al. (2002), CASL models agents as entities with mental states (knowledge and goals). It is based on a declarative action theory defined in the situation calculus (McCartyh and Hayes, 1969) combined to a programming language ConGolog (De Giacomo et al., 2000). CASL models Knowledge using a possible worlds account adapted to the situation calculus. A situation represents a snapshot of the domain. $K(a, s', s)$ is used to denote that in situation s, agent a thinks that it could be in situation s'. $\phi[s]$ means that ϕ is true in the situation s. Using K, the knowledge of an agent is defined as follows:

$$Know(a, \phi, s) =_{def} \forall s' (K(a, s', s) \Rightarrow \phi[s'])$$

An agent a knows a formula ϕ, if ϕ is true in all K-accessible situation by agent a.

In CASL, three variants of the *inform* communicative action are supported (Lespérance, 2002):

$inform(a, b, \phi)$: agent a inform agent b that ϕ currently holds.
$informWhether(a, b, \phi)$: agent a inform agent b about the current truth value of ϕ.
$informRef(a, b, \phi)$: agent a inform agent b of who/what ϕ is.

The preconditions of these three actions are expressed using *Know* predicate. For example, an agent a can inform an agent b that ϕ, iff a knows that ϕ currently holds, and does not believe that b currently knows the truth value of ϕ.

In CASL, goals are modeled using an accessibility relation W over possible situations. The goal accessible situations for an agent are the ones where it thinks that all its goals are satisfied. W-accessible situations may include situations that the agent thinks are impossible. Intentions are defined using *W* and *K* relations so that the intention accessible situations are *W*-accessible situation that are also compatible with what the agent knows, in the sense that there is a *K*-accessible situation in the history of *W*-accessible situations. Thus, unlike goals, agents can only intend things that they believe are possible.

Using the CASL framework, Khan and Lespérence (2004) defined a model of *cooperative ability*, and show how agents use their intentions to determine their next actions. In a single agent domain, an agent's ability to achieve a goal can be defined as its knowledge of a plan that is physically and epistimically executable and whose execution achieves the goal. As argued by the authors, modeling multi-agent ability is more complex because it requires to take into account the agents' knowledge about other's knowledge and intentions as well as how they select actions, behave rationally, etc. At the communication level, the authors extended CASL by providing two intention transfer communication actions: *request* and *requestAct*, and two cancellation actions: *cancelRequest* and *cancelRequestAct*. Finally, they defined *rational plans* and specified a planning framework for cooperating and communicating agents. The main idea in this framework is the role of intention and rationality in adopting a rational plan and in determining an agent's actions.

On the basis of the rational interaction theory, a broad range of ACL performatives have been defined (Huber et al., 2001) (Huber et al., 2004) (Kumar et al., 2000). However, the complexity of the definitions causes sometimes confusion when selecting the correct performative in multi-message exchanges. In addition, these definitions have changed to match changes in the first version of performatives that have been defined, but not all performatives previously defined have been updated with each underlying definition change (Huber et al., 2004).

Several approaches have been defined for implementing cognitive concepts (Huhns and Singh, 1998). According to one of these approaches, the agent represents its beliefs, intentions, and desires in modular data structures and performs explicit manipulations on those structures to carry out means-ends reasoning or plan recognition. When the cognitive concepts are defined formally, the explicit manipulations can be accomplished through the application of a suitable theorem prover. Among the best of the systems using this approach is ARTIMIS (Sadek et al., 1997). ARTIMIS is an intentional system designed for human interaction and applied in a spoken-dialogue interface for information access. This system is based on a logic of beliefs and intentions defined from the Cohen and Levesque framework. In ARTIMIS, agents' communicative acts are modeled as rational actions. The rational unit of the system enables agents to reason about knowledge and plans pertaining to their communicative acts.

One of the other best-known formalizations in the mental approach is Rao and Georgeff's BDI-logic (Rao and Georgeff, 1991). Dealing with desires and intentions as primitives, the authors focus on the process of intention revision. The BDI-architecture is particularly interesting because it combines three distinct components: A philosophical foundation, a software architecture and a logical formalization (van der Hoek and Wooldridge, 2003). Syntactically, BDI logic is essentially branching time logic enhanced with additional modal operators: *Bel*, *Des* and *Intend* to capture agents' beliefs, desires and intentions respectively.

The semantics that Rao and Georgeff give to BDI modalities in their logic are based on Kripke structures and possible worlds. However, rather than assuming that worlds are instantaneous states of the real world, it is assumed that worlds are themselves branching temporal structures. While this enables the authors to define some interesting properties, it complicates the semantic machinery of the logic.

Although Rao and Georgeff's BDI-logic shares much in common with Cohen and Levesque's intention logic, there are two main differences between these two logics. The first and most obvious distinction is that Rao and Georgeff's BDI-logic uses explicitly a CTL-like branching time logic. The second distinction is that worlds are a discrete sequence of events in the formalism proposed by Cohen and Levesque, and are branching temporal structures in the formalism proposed by Rao and Georgeff. In term of expressivity, Rao and Georgeff's approach explores the possible interrelationships between beliefs, desires, and intentions from the perspective of semantic characterization. The most obvious relationships that can exist between agent's belief, desire, and intention accessibility relations are whether one relation is a subset of another. For example, if desire accessibility relation is a subset of intention accessibility relation for a given agent, then we would have as an interaction axiom the fact that if this agent intends that a proposition is true, then it desires that this proposition is true.

Another important formalization is the KARO framework (for Knowledge, Actions, Results and Opportunities) proposed by (van Linder et al., 1998). KARO is a formal system that may be used to specify, analyze, and reason about the behavior of rational agents. The core of KARO is a combination of epistemic and dynamic logic. The framework comes with a sound and complete axiomatization. For instance, it is possible to model, using this framework, that an agent knows that some action is able to bring about some state of affairs since it knows that an action is feasible in the sense that the agent knows of its ability to perform the action.

The main difference between the KARO framework and Cohen and Levesque's approach is that the KARO framework employs explicitly dynamic logic, a programming logic with explicit reference to actions (programs) within the language. In addition, according to Cohen and Levesque's approach, an agent intends to do an action if it has the persistent goal to have done the action, however, in the KARO framework, intentions are represented by commitments consisting of actions. Because commitments have a very computational flavor, The KARO framework is more computational in nature. On the other hand, the difference between Rao and Georgeff' logic and The KARO formalism is that the first logic focuses on the process of intention revision rather than the commitment acquisition which is essential to the KARO framework. Another difference is that BDI-logic rests on temporal logic rather than dynamic logic as in the case of the KARO formalism. Consequently, desires and intentions in BDI-Logic suffer from the problems associated with logical omniscience. A detailed description of these problems is discussed in (Meyer et al., 1999).

Several researchers used the approaches of Cohen and Levesque, Rao and Georgeff or KARO to define a formal semantics of ACLs (Hindriks et al., 2000), (Labrou, 1997), (Labrou and Finin, 1998), (Sadek, 1991), (van Eijk, 2000). For example, according to the semantics proposed by Labrou and Finin (1998), the fact that an agent Ag_1 *informs* another agent Ag_2 that a proposition *p* is true is interpreted as "Ag_1 *believes* that *p* is true and *believes* that Ag_2 *intends* to find whether *p* is true or not". However, these semantics have been criticized for not being verifiable because it is not possible to verify whether the agents' behaviors match their private mental states (Dignum and Greaves, 2000), (Singh, 2000).

4.2.4 Discussion

The mental approach has the advantage of being formally defined on the basis of modal logic and of a logic of action, which explains its success in the field of the human-machine interfaces. It also has the advantage of offering a complete theory which makes it possible to cover the three basic elements of the communication: syntax, semantics and pragmatics which is captured by the concept of planning. However, the approach based on planning has several limitations. The concept of plan can be useful when we consider simple conversations that agents can plan in advance. But as soon as the conversations become more complicated, this approach becomes inadequate. This is due to the fact that the dialogue is a very dynamic activity, whereas plans, although they can be revised when circumstances change, are static in nature because all communicative acts are planed in advance. In addition, plan revision is a computationally complex task. Moreover, the computational complexity of plan recognition algorithms is another limit. The plan recognition problem is also non decidable in certain cases (Bylander, 1991).

The semantics defined in this approach rests on a multimodal logic combined with an action theory. To use a language based on this semantics, agents must be specified according to a BDI approach. This semantics is simple, declarative and unambiguous. However, it remains difficult to verify it because agent's mental states are private. Moreover, this semantics supposes that agents are sincere and cooperative. Although it is useful in certain cases, this assumption is not valid for all dialogue types, for example negotiation and persuasion. In addition, this semantics gives only the meaning of individual performatives and no semantics is defined for conversations. Defining pre / post-conditions of speech acts does not specify how BDI agents can take part in coherent conversations.

4.3 The Social Commitment-based Approach

An alternative to the mental approach was proposed by Singh (1998) and Colombetti (2000) under the name of social approach. In opposition to the mental approach, this approach stresses the importance of conventions and the public and social aspects of dialogue. It is based on social commitments that are thought of as social and deontic notions. As argued by Dignum and her colleagues (Dignum et al., 2003), deontic concepts are important and fundamental elements to specify interactions in agent societies. Social commitments are commitments towards the other members of a community (Castelfranchi, 1995). They differ from the agent's internal psychological commitments which capture the persistence of intentions as specified in the rational interaction theory (Cohen and Levesque, 1990). A speaker is committed to a statement when he made this statement or when he agreed upon this statement made by another participant. In fact, we do not speak here about the expression of a belief, but rather about a particular relationship between a participant and a statement. What is important here is not that an agent agrees or disagrees upon a statement, but rather the fact that the agent *expresses* agreement or disagreement, and acts accordingly. A social commitment is therefore a public attitude of a participant relative to a proposition.

4.3.1 Singh et al.'s Work

This notion of social commitment was proposed in order to define a formal semantics that is verifiable (Singh, 2000). Thus, based on Habermas's work (Habermas, 1984), Singh proposed a three-level semantics such that each act is associated with three validity claims: the objective claim (that the communication is true), the subjective claim (that the communication is sincere) and the practical claim (that the speaker is justified in making the communication). For instance, by *informing* agent *B* that proposition *p* is true, agent *A* (called *debtor*) *commits* towards *B* (called *creditor*) that *p* holds (objective conclusion), that it believes that *p* is true (subjective conclusion), and to the whole agent group that it has a *reason* to believe that *p* is true (practical conclusion). Singh's approach is based on the mental approach when considering the subjective claim which is embedded within a social attitude when considering the practical claim. The practical claim actually leads to a social commitment made by the speaker towards the whole agent group. The commitment-based semantics has therefore been introduced in order to capture these three levels.

Technically, Singh defined the semantics of social commitments as an operator using Computation Tree Logic (CTL) (Emerson, 1990). This semantics is given relative to the following model: $M = <S, <, \approx, N, R, A, B, I, C>$. S is a set of states, $< \subseteq S \times S$ is a partial order indicating branching time, $\approx \subseteq S \times S$ relates states to similar states, $N : S \rightarrow 2^{\Phi}$ is an interpretation which tells us which atomic propositions (Φ) are true in a given state. The set of paths derived from $<$ is denoted P. $R : S \rightarrow P$ gives the real path originating from a state. A is a set of agents. $B : S \times A \rightarrow 2^{S}$, $I : S \times A \rightarrow 2^{P}$, and $C : S \times A \times A \rightarrow 2^{P}$ give the modal accessibility relations for beliefs, intentions, and commitments respectively. B assigns to each agent at each moment the set of moments that the agent believes possible at that moment. I assigns to each agent a set of paths that the agent is interpreted as having selected or preferred. C assigns to each agent a set of paths on which the agent commits towards another agent. A commitment is denoted $Com(Ag_1, Ag_2, p)$ where Ag_1 and Ag_2 are two agents, and p is a propositional formula. The meaning of a commitment is given by the following formula:

$$M \models_t Com(Ag_1, Ag_2, p) \textit{ iff } (\forall pa : pa \in C(t, Ag_1, Ag_2) \Rightarrow M \models_{pa,t} p)$$

$M \models_t p$ expresses "M satisfies p at state t" and $M \models_{pa,t} p$ expresses "M satisfies p at state t along path pa".

Although it is verifiable at the objective level, this semantics remains unverifiable at the subjective level because this level is expressed in terms of mental states. In addition, the semantics given for the notion of social commitments does not reflect the deontic or the public aspect but only the fact that the content is true in the accessible states along some paths. The algebraic properties of this relation are also not specified.

Using Singh's approach, Mallya et al. (2004) defined some constraints in order to capture some operations on commitments. These operations are: *Create* (that establishes the commitment), *Cancel* (that cancels the commitment), *Release* (that releases the debtor from a commitment), *Assign* (that replaces a commitment's creditor by another), *Delegate* (that replaces the commitment's debtor by another), and *Discharge* (that fulfills the commitment). An example of the defined constraints is: a commitment cannot be created more than once with a given identifier. The authors developed a representation for the temporal content capable of capturing realistic contracts. Then, they dealt with the problem of solving temporal commitments by showing how the satisfaction or breach of a commitment can be detected.

On the basis of the social commitment approach, Yolum and Singh (2002) proposed an approach for specifying protocols in which the content of the actions is captured through agent's commitments. In this approach, commitments are formalized using a variant of the event calculus (Kowalski, 1986). The authors used the same operations specified in (Mallya et al., 2004). Then, they defined reasoning rules to capture the evolution of commitments through the agents' actions. Using these rules in addition to the event calculus axioms and an event calculus planner (Shanahan, 2000), agents can reason about their actions. The event calculus planner is used to demonstrate how possible transitions can be generated between an initial state and a goal state given a protocol specification. As a related work, Chopra and Singh (2004) proposed a commitment-based formalism called *non-monotonic commitment machines* for representing multi-agent interaction protocols. This formalism uses commitments for representing states and actions. The meaning of a state is given by the commitments that hold in this state. The meaning of an action is defined by the way it manipulates commitments. This formalism does not directly specify sequences of states and transitions. Instead, it specifies rules in *nonmonotonic causal logic* (Giunchiglia et al., 2003). These rules model the changes in the state of a protocol as a result of the execution of actions. The inference mechanism in this logic computes new states at runtime. The nonmonotonic causal logic is used only to reason about actions in the sense that an action can be the cause for a formula to be true, for example $Create(Ag_1, Ag_2, p)$ causes $Com(Ag_1, Ag_2, p)$ and $Discharge(Ag_1, Ag_2, p)$ causes $\neg Com(Ag_1, Ag_2, p)$.

4.3.2 Colombetti et al.'s Work

Colombetti (2000) proposed a commitment-based semantics for an ACL called Albatross (Agent Language Based on a TReatment Of a Social Semantics). The definition of this ACL is based on an extended first order modal language *L*. This language contains terms of different sorts including: *agent*, *action token*, *action type*, *force indicator*, and *message body*. Colombetti used this language to define the meaning of speech acts according to Searle and Vanderveken's classification (1985). To express the meaning of directive speech acts, he introduced the notion of *precommitment*. For example, when an agent A requests another agent B to do something, A is trying to induce B to make a commitment. In this situation, we speak about a precommitment of A. An expression of the form $C(e, A, B, \varphi)$

(respectively $PC(e, A, B, \varphi)$) means that action e commits (respectively precommits) agent A to φ relative to agent B. If e is an action token and α is an action type, then $Act(e, \alpha)$ means that e is a token of action type α. If A is an agent and e is an action token, an expression of the form $Done(e, A)$ means that agent A has just completed the execution of the action token e. This predicate can be overloaded as follows:

$$Done(e, A, \alpha) =_{def} Act(e, \alpha) \wedge Done(e, A)$$

In Albatross, A message is an expression with sub-expressions specifying a *sender*, a list of *receiver*s, a *force indicator* (in the sense of speech act theory), and a *body* (i.e., a statement of a *content language* conveying the content of the message). If A and B are agents, f is a force indicator, and s is a message body, the term $Send(A, B, f, s)$ denotes the following action type: a message is sent with sender A, B as one of the receivers, force indicator f, and body s. For every message body s there is a logical statement φ such that $Holds(s) \leftrightarrow \varphi$ is valid, where the intuitive meaning of $Holds(s)$ is that s holds. This assumption is considered as meta-theoretic. A term of the form $SpeechAct(A, B, f, \varphi)$ denotes the following action type: a speech act is performed with A as the speaker, B as one of the addressees, force f, and content φ. The speaker of a speech act coincides with the agent that performs it. The relationship between messages and speech acts is expressed through an inference rule:

$$\frac{Holds(s) \leftrightarrow \varphi}{Act(e, Send(A, B, f, s)) \rightarrow Act(e, SpeeechAct(A, B, f, \varphi))}$$

Using the language *L*, Colombetti defined a number of speech acts: *declarations*, *assertives*, *commisives*, *directives*. For example, for an assertive act, the point is to commit its actor to the truth of what is asserted, relative to every addressee and for a directive act, the point is to have the addressee perform some action. Assertive and directive acts are defined as follows:

$$Assert(A, B, \varphi) = SpeechAct(A, B, Assert, \varphi)$$
$$Done(e, A, Assert(A, B, \varphi)) \rightarrow C(e, A, B, \varphi)$$
$$\mathrm{Re}\,quest(A, B, \varphi) = SpeechAct(A, B, \mathrm{Re}\,quest, \varphi)$$
$$Done(e, A, \mathrm{Re}\,quest(A, B, Done(B, \alpha))) \rightarrow PC(e, B, A, Done(B, \alpha))$$

Fornara and Colombetti (2002) defined an operational specification of Albatros by using social commitments. The essential components of this specification are: a commitment class that can be instantiated to a set of commitment objects, a fixed set of actions that agents may perform and a fixed set of roles that agents play during an interaction. Some basic operations on commitments are defined: *Make commitment*, *Make precommitment*, *Cancel commitment*, *Cancel precommitment*, *Accept precommitment*, *Reject precommitment*. These operations are used to define the meaning of the basic types of communicative acts as identified by speech act theory. The authors used this specification to define some interaction protocols (Fornara and Colombetti, 2003, 2004).

Verdicchio and Colombetti (2003) proposed a logical model of social commitments based on CTL^{+-} (CTL* augmented with past operators). The purpose of their framework is to define an ACL semantics based upon the concept of social commitments. This framework relies on the assumption that agent communication should be analyzed in terms of communicative acts, by means of which agents create and manipulate commitments. They extended the temporal language of CTL^{+-} in order to represent events and actions. Events are treated as a sort of individuals called *event tokens*. Every event token belongs to at last one *event type*, and takes place (*happens*) at exactly one time instant. By taking $Happens(e), Type(e,t)$ and $Actor(e,x)$ as primitives, with e is an event token, t an event type, and x an agent, they defined:

$$Done(e,x,t) =_{def} Happens(e) \wedge Type(e,t) \wedge Actor(e,x)$$

This formula expresses the fact that event e of type t is brought about by agent x. Commitments and precommitments are only defined syntactically by two predicates $Comm$ and $Prec$ without any semantics. $Comm(e,x,y,u)$ (respectively $Prec(e,x,y,u)$) means that event e has brought about a commitment (respectively a precommitment) for agent x, relative to agent y, to the truth of u. The action types for commitment and precommitment manipulation are defined by axioms describing their constitutive effects, that is, by describing the state of affairs that necessarily hold if a token of a given action type is successfully performed. For example, the following axiom says that: if an agent x successfully performs an action of making a commitment with x as the debtor, y as the creditor, and u as the content, then on all paths agent x is committed, relative to y, to content u, until agent x possibly cancels such a commitment, after which the commitment no longer exists. The authors also studied fulfillment and violation of commitments.

Using commitment-based semantics proposed by Colombetti (2000) and by Verdicchio and Colombetti (2003), Fornara, Vigano, and Colombetti (2004) proposed to regard an ACL as a set of conventions to act on a fragment of institutional reality. Communicative acts are regarded as a sort of institutional actions, that is, as actions performed within an institution to modify a fragment of social reality (Searle, 1995). According to the authors, defining the semantics of an ACL has two sides: one side is the definition of the institutional effects brought about by the performance of communicative acts; the other side is the definition of the social context in which agents can carry out institutional actions. Institutional actions are particular types of actions that agents cannot perform by exploiting causal links. Rather, institutional actions are performed on the basis of a shared set of conventions and norms. Norms prescribe which institutional actions should or should not be executed among those that are authorized. They are important in the sense that they make an agent's behavior at least partially predictable and allow agents to coordinate their actions according to the expected behavior of the others.

The approach proposed by Colombetti, Fornara, Verdicchio, and Vigano offers an operational specification and a logical definition of agent communication. However, this approach is only based on the notion of social commitments and it neglected the agents' mental states and their reasoning process. Without this process it is not clear how agents manipulate their commitments when conversing.

4.3.3 Flores et al.'s Work

Flores and Kremer (2002) proposed a social model for agent conversations for action based on social commitments and their negotiation. They used observable behavior and the concept of shared social commitments to ensure the coherence of agent conversations. During the conversation, each agent maintains a private record to which shared commitments are added and from which they are removed. The authors formally specify their model using the Z language.

In addition, they defined a basic protocol for the negotiation of social commitments called PFP (Protocol For Proposals). The protocol starts with a proposal from a sender to a receiver to concurrently adopt or discharge a social commitment. Either the receiver replies with an acceptance, rejection, or counteroffer or the sender issues a withdrawal or counteroffer. All utterances except a counteroffer terminate an instance of the protocol. Finally, it is expected that when an acceptance is issued, both speaker and addressee will simultaneously apply the proposed commitments to their record of shared commitments.

Flores et al. (2004) presented a conversational model where the meaning of messages is based on their use as coordinating devices. They distinguished two types of meaning: *speaker's meaning*, which is based on the use of messages for the communication of intent, and *signal meaning*, which is based on the use of messages as coordinating devices incrementing the common ground of interacting agents. Following this view, the meaning of messages is incrementally defined based on the following levels: a *compositional level*, where the meaning of messages is given according to their constituents; a *conversational level*, where the meaning of messages is given based on their occurrence as part of a conversation in which agents concur to advance the state of commitments; a *commitment state level*, where the meaning of messages is given according to the state of the commitments these messages manipulate; and a *joint activity* level, where the meaning of messages is given according to their use in joint activities.

4.3.4 Discussion

The social approach is regarded as a change in agent design: from individual representation (private representation) to social interaction (public representation). An ACL must be conceived taking certain standards into consideration in such a way that agents belonging to different environments could interact. These standards are supposed to provide the possibility of testing the compliance of these agents with respect to the ACL specification. Commitment-based semantics has the advantage of being verifiable because unlike mental states, commitments are objective and public. They do not require to be reconstituted using inference processes. Compliance testing in this approach is based on the following idea: an observer of a MAS can maintain a record of the commitments being created and modified.

From these, the observer can determine the compliance of other agents with respect to the given protocol. However, this technique does not allow us to check whether the protocol satisfies or not the properties that it should satisfy and whether the participating agents respect or not the semantics of the communicative acts. Indeed, when agents communicate using a semantics, we need to verify that they use the same semantics. In Chapter 8, we address this problem in a formal way using a model checking technique.

This approach has also been critiqued in (Khan and Lespérance, 2004) because communication cannot be reduced to the public social commitments level. The reason agents communicate is that this serves their private goals. Therefore, they must reason about these goals and the associated beliefs when communicate. Thus, a mentalistic semantics is also essential. For this reason, we think that a combined mental-social-argumentative semantics provides a good understanding of the agents' communicative behavior.

On the other side, specifying protocols using a commitment-based approach does not provide a solution to the flexibility problem if agents cannot reason about their commitments. Although the event calculus planner and causal logic offer a reasoning mechanism to agents, this reasoning remains elementary. The reason is that agents cannot decide about the next act to be performed. The decision-making process is not taken into account in the protocols suggested in this approach. In Chapters 5 and 6, we show that using an argumentative theory in this approach provides such a process. On the other hand, in Chapter 9, we show that integrating dialogue games in a hybrid approach based on commitments and arguments provides more flexibility for these protocols.

The approach proposed by Colombetti and his colleagues is completely based on the social commitments and neglects the agents' mental aspect. Therefore, this approach captures only the observable part of the communication, and does not explain how agents can participate in conversations. Finally, although the approach proposed by Singh mentions agents' mental states, it does not specify how agents establish the link between their mental states and the different commitments. For example, how agents handle their commitments on the basis of their mental states is not specified. In our pragmatic approach (Chapters 5 and 6), we show how this link is established using the agents' reasoning mechanism.

4.4 The Argumentative Approach

Another approach, called the argumentative approach, was proposed by Amgoud and her colleagues (Amgoud, 1999), (Amgoud et al., 2000a, 200b, 2002) as an extension to Dung's work (Dung, 1995), and by McBurney and his colleagues (McBurney and Parsons, 2000), (McBurney, 2002), (McBurney et al., 2002). This approach is based upon an argumentation system that can include a preference relationship between arguments (Amgoud, 1999). According to this approach, the agents' reasoning capabilities are often linked to their ability to argue. They are mainly based on the agent's ability to establish a link between different facts, to determine if a fact is acceptable, to decide which arguments support which facts, etc. Before studying this approach we introduce some preliminary concepts.

4.4.1 Preliminary Concepts

Argumentation theory has been applied in the design of intelligent systems in several ways over the last decade. Arguments can be considered as tentative proofs for propositions (Fox et al., 1992), (Krause et al., 1995). One may imagine that knowledge in some domain is expressed in a logical language, with the axioms of the language corresponding to premises in the domain. Theorems in the language correspond to claims in the domain which can be derived from the premises by successive applications of some set of inference rules. For many real-life domains, the premises will be inconsistent in the sense that contrary propositions may be derived from them. In this formulation, arguments for propositions, or claims, are the same as proofs in a deductive logic, except that the premises on which these proofs rest are not all known to be true. Arguments are thus treated as *tentative proofs* for claims.

Many formalisms of argumentation such as (Pollock, 1991, 1992), (Prakken and Sartor, 1996), and (Vreeswijk, 1997) regard an argument as a structured chain of rules. An argument begins with one or more premises. After this follows the repeated application of various rules, which generate new conclusions and therefore enable the application of additional rules.

The understanding of an argument as a tentative proof and a chain of rules attends to its internal structure, as analogous to a chain of inference steps connecting a set of premises to a claim. A second strand of research in artificial intelligence has emphasized the relationship between arguments when considered as abstract entities, ignoring their internal structures. This approach has enabled argumentation systems to be defined as *defeasible* reasoning systems (Pollock, 1991, 1992), (Simari and Loui, 1992). Arguments are thus defeasible, meaning that the argument by itself is not a conclusive reason for the conclusions it brings about. In defeasible logic (also called nonmonotonic logic), inferences are defeasible, that is, the inferences can be defeated when additional information is available.

In this logic, the conclusions are not deductively valid: it is possible that the premises are true while the conclusion is not. Whether or not an argument should be accepted depends on its possible counterarguments. To decide about the acceptability of arguments, Dung (1995) proposed the use of a formal argumentation framework. In this framework, an *argument framework* is a set of arguments (considered as abstract entities) together with a binary relationship across this set, called *attack*. A set of arguments *Args* is *conflict-free* if there is no arguments Arg_1 and Arg_2 in *Args* such that Arg_1 attacks Arg_2. Any given argument is said to be *acceptable* with respect to a designated subset *S* of the set of arguments if every argument which attacks the given argument is itself attacked by an argument in the designated subset. Such a subset *S* is said to be admissible if it is conflict-free and if every argument it contains is acceptable with respect to *S*. Intuitively, acceptable arguments with respect to some set S are those which are defended by the elements of S against all attacks. Similarly, an admissible set of arguments is one which defends its own members against all attacks.

4.4.2 Dialectical Models of Argumentation

The monological models of argumentation, like Toulmin's model (Toulmin, 1958), focus on structural relationships between arguments. On the contrary, formal dialectics proposes dialogical structures to model the connectedness of utterances. Dialectical models focus on the issue of fallacious arguments, i.e., invalid arguments that appear to be valid. They are rule-governed structures of organized conversations in which two parties (in the simplest case) speak in turn in an orderly way. These rules are the principles that govern the participants' acts, and consequently the use of dialectical moves.

Hamblin (1970) and MacKenzie (1979) proposed a mathematical model of dialogues. They defined some connectors necessary to the formalization of the propositional contents of utterances, and a set of locutions for capturing the speech acts performed by participants when conversing. The dialectical system proposed by MacKenzie, and called *system DC*, is an extension to the one proposed by Hamblin. MacKenzie's DC proposed in the course of analyzing the fallacy of question-begging provides a set of rules for arguing about the truth of a proposition. Each participant, called player, has the goal of convincing the other participant, and can assert or retract facts, challenging the other player's assertions, ask whether something is true or not, and demand that inconsistencies be resolved. When a player asserts a proposition or an argument for a proposition, this proposition or argument is inserted into a public store accessible to both participants. These stores are called *commitments stores* (*CS*). There are rules which define how the commitment stores are updated and whether particular illocutions can be uttered at a particular time.

A MacKenzie's dialectical system mainly consists of:
1. A set of moves: they are linguistic acts, for example assertions, questions, etc.
2. A commitment store: it contains the different propositions and arguments asserted by the players. This store, accessible by all the players, makes it possible to keep the trace of the various phases of the dialogue.
3. A set of dialogue rules: they define the allowed and the prohibited moves. These rules have the following form "if condition, moves *C* are prohibited". A dialogue is said to be *successful* when the participants conform to its rules.

The language used in DC contains propositional formulae: "p", "$\neg p$" and "$p \vee q$". Locutions are constructed from *communicative functions* that are applied to these propositions. For example, the moves: "question(fine)" and "assertion (fine, fine $\rightarrow$ hot)" indicate respectively the question "is it fine?" and the assertion "the weather is fine, and when the weather is fine, the weather is hot".

Table 4.1 illustrates the evolution the CSs of two players *A* and *B* during the following dialogue:

A1: *The doctors cannot make this surgery*
B2: *Why*?
A3: *Because the patient is too old and that he refuses*
B4: *Why does he refuse*?

A5: *Because there is little chance of success.*

Turn	Player	Move	CS(*A*)	CS(*B*)
1	*A*	Assert(¬d)	¬d	¬d
2	*B*	Challenge(¬d)	¬d	?¬d
3	*A*	Assert(p ∧ ¬a)	¬d, p ∧ ¬a, p ∧ ¬a → ¬d	?¬d, p ∧ ¬a, p ∧ ¬a → ¬d
4	*B*	Challenge(¬a)	¬d, p ∧ ¬a, p ∧ ¬a→ ¬d	?¬d, p, ?¬a, p ∧ ¬a → ¬d
5	*A*	Assert(s)	¬d, p ∧ ¬a, s, p ∧ ¬a → ¬d, s → p	?¬d, p, ?¬a, s, p ∧ ¬a → ¬d, s → p

Table 4.1. The evolution of CSs during a dialogue

The dialogue starts with *A*'s assertion (¬*d*): "the doctors cannot make this surgery". Thus, *A* commits itself and commits its adversary *B* to this fact. Thereafter, *B* challenges this assertion (one speaks in this case about a disengagement on the fact and an engagement on the challenge). After that, *A* provides a justification, which commits the two players to this assertion and to the fact that this assertion logically implies the challenged fact. The dialogue continues in a similar way with *B*'s challenge of an *A*'s justification part, which involves a new *A*'s justification.

4.4.3 Modeling Dialogue using Argumentation

Several researchers have attempted to use argumentation techniques for modeling and analyzing negotiation dialogues (Sycara, 1990), (Parsons and Jennings, 1996), (Tohmé, 1997) (Rahwan et al., 2004). Amgoud and her colleagues (2000a, 2000b) extended these proposals by investigating the use of argumentation for a wider range of dialogue types. In this section we summarize this work.

The approach proposed by Amgoud et al. relies upon MacKenzie's formal dialectics. The dialogue rules of this system are formulated in terms of the arguments that each player can construct. Dialogues are assumed to take place between two agents, *P* and *C*, where *P* is arguing in favor of some proposition, and *C* argues "con". Each player has a knowledge base Σ_P and Σ_C respectively, containing their beliefs. As in DC, each player has another knowledge base, accessible to both players, containing commitments made during the dialogue. These commitment stores are denoted *CS*(*P*) and *CS*(*C*) respectively. The union of the commitment stores can be viewed as the state of the dialogue at turn *t*. All the bases described above contain propositional formulae and are not closed under deduction.

Both players are equipped with an argumentation system. Each has access to his own private knowledge base and to both commitment stores. The two argumentation systems are then used to help players to maintain the coherence of their beliefs, and thus to avoid asserting things which are defeated by other knowledge from $CS(P) \cup CS(C)$. In this sense the argumentation systems help to ensure that players are *rational*.

To model dialogue types proposed by Walton and Krabbe (1995) (see Chapter 2, Section 2.7.2), the authors used seven dialogue moves: *assert*, *accept*, *question*, *challenge*, *request*, *promise* and *refuse*. For each move, they defined *rationality rules*, *dialogue rules*, and *update rules*. The rationality rules specify the preconditions for playing the move. The update rules specify how commitment stores are modified by the move. The dialogue rules specify the moves the other player can make next, and so specify the *protocol* under which the dialogue takes place. Figure 4.3 presents these rules for the *assert* and *challenge* moves.

The authors showed that this framework can be used to implement the language for persuasive negotiation interactions proposed by Sierra et al. (1998). In (Parsons et al., 2002), this approach is used to analyze formal agent dialogues using the dialogue typology proposed by Walton and Krabbe. The authors defined a set of locutions by which agents can trade arguments and a set of protocols by which dialogues can be carried out. In (Parsons et al., 2003), this approach is used to examine the outcomes of the dialogues an argumentation system permits. As an outcome, the authors used the set of acceptance propositions (i.e. what agents come to accept during the course of the dialogue). This argumentation approach has the advantage of linking communication and reasoning as well as of being verifiable. However, the approach by itself does not allow capturing certain notions such as obligations, conventions, roles, etc.

***assert*(*p*)** where p is a propositional formula.

Rationality the player uses its argumentation system to check if there is an acceptable argument for the fact p.

Dialogue the other player can respond with:

1: accept(p)
2: assert($\neg p$)
3: challenge(p)

Update $CS_i(P) = CS_{i-1}(P) \cup \{p\}$ and $CS_i(C) = CS_{i-1}(C)$

***challenge*(*p*)** where p is a propositional formula.

Rationality $\emptyset$

Dialogue the other player can onle assert (S) where S is an argument supporting p.

Update $CS_i(P) = CS_{i-1}(P)$ and $CS_i(C) = CS_{i-1}(C)$

Figure 4.3. Example of rationality, dialogue and update rules

4.4.4 Other Work

On the basis of Amgoud et al.'s work, Sadri et al. (2001) proposed a protocol but with fewer locutions called *dialogue moves*. The legal dialogue moves are *request*, *promise*, *accept*, *refuse*, *challenge* and *justify*. The content of the dialogue moves *request* and *promise* are resources, while the content of the other four dialogue moves are themselves dialogue moves. For example, *accept*(*Move*) is used to accept a previous dialogue move *Move* and *challenge*(*Move*) is used to ask a justification for a previous dialogue move *Move*. Because the intended application is a dialogue over scarce resources, the authors proposed a semantic linking utterances to a first-order logic describing resources. In this framework, an agent's knowledge is described as an abductive logic program consisting of *if then* rules and of the resources owned by the agent. The abducibles of this logic program are the possible locutions which the agent may utter in response to a message it receives.

The research work on argumentation that we have described concentrates on formal dialectics. Another field of argumentation in artificial intelligence focuses on discourses which are *rhetorically* argumentative. This field, called *rhetorical argumentation*, deals with arguments which are both based on the audience's perception of the world, and with evaluative judgments rather than with establishing the truth of a proposition (Grasso, 2002). In Aristotle's rhetorical argumentation, the emphasis is put on the audience rather than on the argument itself. In a persuasive dialogue, the rhetorician appeals to the audience's set of beliefs in order to try to persuade this audience, rather than to achieve general acceptability (Aristotle, 1926). Using Aristotle's definition, philosophers Perelman and Olbrechts-Tyteca (1969) proposed a n*ew rhetoric* theory aiming at identifying discursive techniques. Based on an approach that goes from examples to generalization, this theory proposes a collection of argument *schemas* which are successful in practice. This collection is classified in terms of the objects of the argumentation and the types of audience's beliefs that the schema exploits. Each schema is described by associations of concepts, either known or new to the audience in order to win the audience's acceptance. A rhetorical schema is meant to express when it is admissible to use a given relationship between concepts. Grasso used this theory to propose a framework for rhetorical argumentation (Grasso, 2002) and a mental model for a rhetorical arguer (Grasso, 2003). The purpose is to build artificial agents able to engage in rhetorical argumentation. In this framework, argumentation aims at reaching an evaluation of an object or of a state of affairs. This evaluation is a way to pass value from one topic to another, in the same way as a deductive argument passes truth from one proposition to another. Formally, we say that there exists an evaluation of a concept c, in the set of concepts C from a certain perspective p of a set P from which the evaluation is made, if there exists a mapping E of the pair (c, p) into a set V of values. Assuming that V is a set consisting of two elements: *good* and *bad*, we write:

$$E: C \times P \rightarrow V = \{good, bad\}$$

Grasso defines a rhetorical argument as the act of putting forward the evaluation of a concept, on the basis of a relationship existing between this concept and another concept, and by means of a rhetorical schema. If we have a concept c and an evaluation of such a concept, we can put forward a rhetorical argument in favor or against a second concept c' iff 1) a relationship exists between the two concepts c and c' and 2) a schema can be identified that exploits such a relation.

As a related work, Reed, Walton and Prakken (Prakken et al., 2003), (Reed and Walton, 2003), (Walton and Reed, 2003) proposed a classification and a formalization of *argumentation schemes*. Argumentation schemes are forms of argument (structures of inference) representing common types of argumentation. They represent structures of arguments used in everyday discourse, as well as in specific contexts such as legal argumentation or scientific argumentation. They represent the deductive and inductive forms of argument which are classical in logic. But they can also represent forms of argument that are neither deductive nor inductive, but that fall into a third category, sometimes called abductive or presumptive. The authors illustrated how argumentation schemes should be fitted into the technique of argument diagramming, using an XML system: the Araucaria (Reed and Rowe, 2001). This system provides an interface through which the user can mark up a text of discourse to produce an argument diagram. They also studied how to model legal reasoning about evidence within general theories of defeasible reasoning and argumentation.

4.4.5 Discussion

The advantage of the argumentative approach lies in the link that it establishes between communication and reasoning. Like humans, agents must reason to be able to take part in intelligent dialogues. In addition, the distinction made between the reasoning level (rationality rules) and the commitment level (update rules) is important for the use of an ACL because it makes it possible to show in an implicit way the relation between agent reasoning (in particular on the basis of its argumentation system) and its participation in conversations. However, the commitment level remains elementary since it only captures the propositions asserted in a dialogue. Other commitment types, such as commitments to do actions and conditional commitments are not taken into account. Moreover, the handling of these commitments in a dialogue is only reflected by the addition and the suppression of propositions in or from commitment stores. However, *attack*, *defense*, *justification* and *withdrawal* operations that can be applied to these commitments are not supported. In addition, to accept or refuse arguments, agents must use not only their argumentation systems but also some social considerations such as agents' trustworthiness.

The dialectical systems on which this approach is based has the advantage of being governed by dialectical rules. These systems are normative frameworks of argumentation considered as dialectical games that each agent must win. This winning-based vision is useful for modeling certain argumentative dialogues like persuasion and negotiation. However, it is not adapted for cooperative dialogues like information-seeking or problem resolution dialogues. In fact, although the formal dialectics provides a dialogical structure, it does not offer a complete dialogue model. The reason is that the evolution and the dynamics of dialogues are only captured by their histories presented by the concept of commitment stores. These histories do not represent the dialogue state and do not distinguish the argumentation phases from the other phases.

In addition to these approaches, certain researchers added to the mental approach some social aspects. These combined approaches are called intentional-conventional approaches.

4.5 The Intentional-Conventional Approaches

As outlined by Clark (1974), agent communication is both a cognitive and a social activity. The mere individual dispositions of the participants cannot explain this phenomenon in a satisfactory manner. This is why an increasing number of researchers often use the terms of mixed or reactive / deliberative approaches (Pulman, 1996), (Traum, 1996), (Hulstijn, 2000a). During the conversation, deliberative processes related to the participants' intentions and desires can take place, as well as more reactive processes related to the conventional aspects of the interactions. The idea is to integrate social attitudes (obligations, interpersonal relationships, roles, powers, etc.) into mental approaches.

In this respect, Pulman (1996) introduces a BDIO (Belief-Desire-Intention-Obligation) approach. In the same direction, Broersen and his colleagues proposed the BOID approach (Broersen et al., 2001). This approach is an abstract agent representation that consists of the four components Beliefs, Obligations, Intentions and Desires. The simple-minded BOID is a lightweight stimulus response agent, that only exhibits reactive behavior. This simple-minded BOID is extended (as time and resources allow) with capabilities for deliberation which may result in more complex (e.g. pro-active) behavior. The BOID architecture contains mechanisms to solve conflicts between the outputs of the four components. This approach consists of two phases: the first phase results in an intermediate epistemic state, and the second phase results in new intended actions. Moreover, Rousseau, Moulin and Lapalme (1996) presented a multi-agent system for simulating conversations involving software agents based on a conversation model and communication protocols designed in order to take into account phenomena present in human conversations. The conversation is thought of as a language game (Wittgenstein, 1958) in which agents negotiate about the mental states they transfer to their interlocutors. An agent proposes certain mental states (beliefs, intentions, emotions, etc.) and other participants react to these proposals, accepting or rejecting the proposed mental objects, asking for further information or justifications, etc. Agents *position* themselves with respect to the transferred mental states. In the same direction, Moulin and Bouzouba (Moulin, 1998), (Bouzouba and Moulin, 1999), suggest adding mechanisms enabling agents involved in a conversation to manipulate social knowledge such as the agents' social power within the interaction context. They show that agents' social relationships should be taken into account in the interaction framework. Thus, they propose an architecture (a conversation manager) that stresses the importance of social relationships and allows agents to handle explicit and implicit information conveyed by speech acts.

4.6 Comparison

In this chapter, we reviewed a certain number of proposals relevant for the study of the general problem of communication between software agents in a MAS. These various proposals share the theoretical base provided by speech act theory. Beyond the isolated aspect of exchanges, agents can communicate by using traditional protocols like those of FIPA or those based on dialogue games. Table 4.2 illustrates a comparison between these proposals on the basis of three criteria: formalisms, semantics and pragmatics.

The semantics of the mental approach is unverifiable since it is impossible to check, without access to the agent' programs, the compliance of this agent with respect to the given semantics. For example, if an agent *A* informs another agent *B* that *p* is true, one cannot check whether or not *A* believes that *p* is true. Because it is based on public commitments, the semantics of the social approach is verifiable. The semantics of the argumentative approach is also verifiable because it uses arguments that are public. For example, if an agent *A* informs another agent *B* that *p* is true, one can check whether or not agent *A* has an argument supporting *p* by challenging it. These three semantics are declarative because they are based on attitudes that are described declaratively rather than by procedures. These semantics describe the meaning of the communicative acts rather than how they can be used.

	Formalisms	**Semantics**	**Pragmatics**
Mental approach	BDI logic (temporal + action logic + situation calculus)	Unverifiable, declarative	Planning
Social approach	Commitment logic (temporal logic), Causal logic, Event calculus	Verifiable, declarative	Commitment-based protocols, commitment-based dialogue games
Argumentative approach	Defeasible logic	Verifiable, declarative	Formal dialectics, dialogue games

Table 4.2. A comparison of the mental, the social and the argumentative approaches

At the pragmatic level, the mental approach is based on the concept of planning, whereas the argumentative approach uses the formal dialectics and dialogue games. On the other hand, the social approach uses operational descriptions of protocols specified by commitments.

It is clear that the pragmatic level must be improved because planning, formal dialectics and commitment-based protocols do not allow agents to take part in conversations in a flexible way while respecting their autonomy. In order to participate flexibly in complex conversations such as negotiations, persuasions and deliberations, agents must be able to make decisions and not only to execute pre-defined plans and protocols. In addition, in the research work on agent communication there is no conversational model that specifies the dynamics and the evolution of conversations and that provides an efficient decision making process enabling agents to decide how to act next. On the other hand, the approaches discussed in this chapter do not take into account the social relationships that can exist between agents, for example how agents' trustworthiness can be considered as an acceptability criterion of arguments. Finally, these approaches do not address the correctness and the verification issues of the communication mechanisms. Verifying that a given agent communication protocol satisfies some properties that are important in a given application context, and verifying that agents respect the semantics when communicating are interesting aspects yet to be addressed. In the second part of this dissertation, we propose our unified framework for the pragmatics and the semantics in which we address these different issues.

Chapter 5*

A Pragmatic Approach based on Social Commitments and Arguments

In this chapter, we propose a formal approach for modeling the pragmatics of agent communication. This pragmatics captures the evolution and the dynamics of agent conversations. This approach is based on the combination of the social approach and the argumentative approach. The link between commitments and arguments that we establish in this chapter enables us to capture both the public and the reasoning aspects of agent communication pragmatics. On the basis of this approach we also propose a layered communication model and a conversational agent architecture.

5.1 Introduction

Agent communication pragmatics deals with the way that agents use communicative acts when conversing. Pragmatics is related to the dynamics of agent interactions and to the way of connecting individual acts while building complete conversations. In the domain of agent communication, many researchers addressed pragmatics. For example Dastani and his colleagues (2000), Fornara and Colombetti (2003, 2004) and Pitt and Mamdani (2000) proposed the notion of protocols as a pragmatic mechanism. Pasquier and his colleagues (Pasquier and Chaib-draa, 2003), (Pasquier et al., 2003) proposed a cognitive coherence theory for this pragmatics. However, these approaches do not specify the evolution of conversations and they are specified informally or semi-formally. In addition, protocol-based approaches do not indicate how agents select their communicative acts. In the cognitive coherence approach, this aspect is addressed using the cognitive dissonance theory that enables agents to cognitively react to a statement. However, this approach does not allow agents to argue, for example, in order to persuade another agent or to negotiate with it.

In this chapter, we propose theoretical foundations for an approach to agent communication pragmatics. This approach uses three fundamental elements: social commitments, actions, and arguments. As illustrated in Figure 5.1, these elements are

---

* We would like to thank John-Jules Ch. Meyer, Frank Dignum and Henry Prakken from Utrecht University, Intelligent Systems Group, and Yves Lespérance from York University for their useful comments about the approach presented in this chapter. This approach is published in (Bentahar et al., 2003, 2004c).

separated in three levels. The first level includes social commitments that agents use in their conversations. The second level includes actions that agents apply to the commitments. The speech acts that agents perform when conversing are defined in terms of these actions. The third level is composed of arguments that agents use to support their actions applied to the commitments. The evolution of agent conversations is represented by the notion of commitment state. Agents use their argumentation systems in order to be able to select the appropriate communicative acts to be performed considering the current state of the conversation.

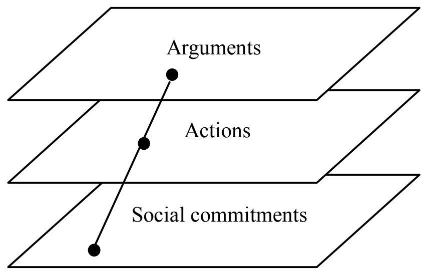


Figure 5.1. The elements of our approach

The purpose of this chapter is to introduce our pragmatic approach that we will use in Chapter 6 in which we develop our framework called commitment and argument network. This framework models the connection between the communicative acts in a conversation and the evolution of this conversation. It is also a means that helps agents to communicate. We also use this pragmatic approach to propose a new persuasion protocol that we develop in Chapter 9.

This chapter is organized as follows. In Section 5.2, we present our social commitment-based framework. In Section 5.3, we introduce the notion of commitment state. A taxonomy of social commitments is given in Section 5.4. In Section 5.5, we establish the link between commitments and arguments. In Section 5.6, we present our communication model. In Section 5.7, we conclude the chapter by a discussion.

5.2 Social Commitments

A social commitment *SC* is a public commitment made by an agent (called the *debtor*), and directed towards a set of agents (called *creditors*) (Castelfranchi, 1995), indicating that some fact is true or that some action will be performed. A commitment is an obligation in the sense that the debtor must respect and behave in accordance with this commitment. A representation of this notion as directed obligations using a deontic logic is proposed in (Herrestad, 1995). Commitments are social in the sense that they are expressed publicly and governed by some rules. This means that they are observable by all the participants. The main idea is that a speaker is committed to a statement when he made this statement or when he agreed upon this statement made by another participant and acts accordingly. What is important here is not that an agent agrees or disagrees upon a statement, but rather the fact that the agent *expresses* agreement or disagreement.

Consequently, social commitments are different from the agent's private mental states like beliefs, desires and intentions. This notion allows us to represent agent conversations as observed by the participants and by an external observant, and not on the basis of the internal agents' states.

In our framework, we distinguish between the social commitment which can be modeled as an object, and the social commitment content. This distinction will be discussed latter in this section. The commitment content is characterized by a time t_φ, which is generally different from the utterance time denoted $t_{u,}$ and from the time associated with the commitment and denoted t_{sc}. t_φ is the time described by the utterance, and thus by the content φ. Time t_{sc} that can be used as an identifier of the commitment refers to the time during which the commitment holds. It can correspond to a fixed value or an interval. When it is an interval, this time is denoted $[t_{sc}^{\inf}, t_{sc}^{\sup}]$. When a temporal bound is instantiated, it takes a numerical value that respects the time unit used by agents. For example, let us consider the following utterance U sent by agent Ag_1 to agent Ag_2: U: I will give you 5$ at 5PM. We can describe the content by the following predicate: $\varphi = Give(Ag_1, Ag_2, 5\$)$. We have: $t_\varphi = 5PM$. The commitment time t_{sc} is an interval: $t_{sc} = [t_u, 5PM]$ with t_u is the utterance time (Figure 5.2).

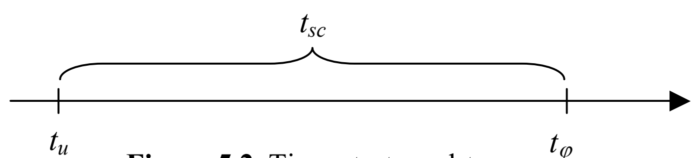


Figure 5.2. Times t_u, t_{sc} and t_φ

If the commitment is satisfied or violated we have $t_{sc} = [t_u, t_\varphi]$. However, if the commitment is withdrawn, we have: $t_{sc} = [t_u, t_w]$, with t_w the withdrawal time (Figure 5.3). Time t_{sc} indicates the time during which the commitment holds, i.e. the time during which the commitment is *active*. Time t_φ indicates the moment at which the commitment must be satisfied.

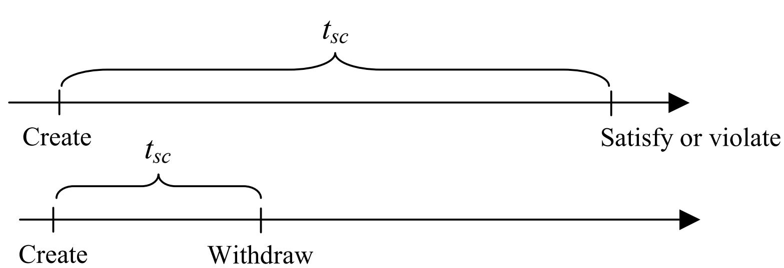


Figure 5.3. Time t_{sc}

We denote a social commitment as follows:

$SC(Ag_1, A^*, t_{sc}, (\varphi, t_\varphi))$

where Ag_1 is the debtor, A^* is the set of the creditors ($A^*=A / \{Ag_1\}$, where A is the set of participants), t_{sc} is the time associated with the commitment, φ its content and t_φ the time associated with the content φ. A social commitment can be identified by t_{sc}. Logically speaking, a commitment is a public propositional attitude. The logical semantics of this notion is defined in Chapter 7. The content of a commitment can be a proposition or an action. A detailed taxonomy of the social commitments that we use in our approach will be discussed latter. To simplify the notation, we suppose throughout this chapter that $A = \{Ag_1, Ag_2\}$.

In order to model the dynamics of conversations, we interpret speech acts as *actions* performed on commitments. A speech act is an abstract act that an agent, *the speaker*, performs when producing an utterance U and addressing it to another agent, *the addressee*. According to the Speech Act Theory (Searle, 1969), (Searle and Vanderveken, 1985), the primary units of meaning in the use of language are not isolated propositions but rather speech acts of the type called *illocutionary acts*. Assertions, questions, orders and declarations are examples of these illocutionary acts. For the moment, our interpretation of a speech act can be denoted by:

$$SA(i_k, Ag_1, Ag_2, t_u, U) =_{def} Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$$

where $=_{def}$ means "is interpreted by definition as".

The definiendum ($SA(i_k, Ag_1, Ag_2, t_u, U)$) is defined by the definiens ($Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$) as an action performed on a social commitment. SA is the abbreviation of "Speech Act", i_k is the identifier of the speech act, Ag_1 is the speaker, Ag_2 is the addressee, t_u is the utterance time, U is the utterance and Act indicates the action performed by the debtor on the commitment: $Act \in \{Create, Withdraw, Reactivate, Violate, Satisfy\}$.

These five actions are the actions that the debtor can apply to a commitment and reflect only the debtor's point of view. However, we must also take into account the creditor when modeling a conversation which is, by definition, a joint activity. The following example illustrates this aspect:

U1: *Quebec is the capital of Canada.*
U2: *No, the capital of Canada is Ottawa.*

The utterance *U1* leads the debtor to create a commitment whose content is "Quebec is the capital of Canada". On the other hand, the utterance *U2* highlights a creditor's action on this content that is in this case a refusal. We thus propose to model the creditors' actions, which are applied to the commitment contents and not to the commitments themselves (Figure 5.4). This separation between the commitment and its content enables us to remain compatible with the semantics of commitments, i.e. the fact that only the debtor can handle its commitments. The creditor can only handle the content of the debtor's commitment. Hence, we must differentiate between the actions applied to a commitment *Act* and the actions performed on the content of a commitment *Act-content*: *Act-content* $\in$ {*Submit-content*, *Accept-content*, *Refuse-content*, *Challenge-content*, *Change-content*, *Suspend-content, Justify-content*, *Defend-content*, *Attack-content*}. We denote an action applied to the content of an Ag_i's commitment as follows:

$Act\text{-}content(Ag_k, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))$

where $i, j, k \in \{1, 2\}$ and $i \neq j$.

Agent Ag_k can thus act on the content of its own commitment (in this case we get $k = i$) or on the content of another agent's commitment (in this case we get $k = j$).

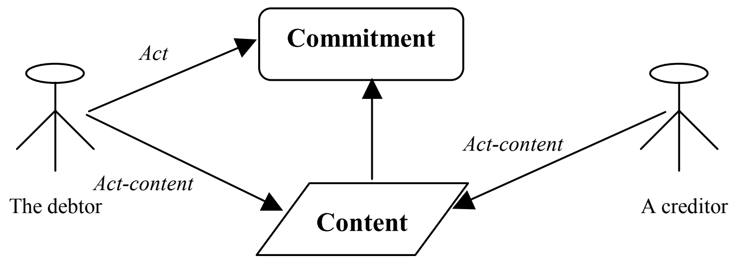


Figure 5.4. Debtors' and creditors' actions

Thus, a speech act leads either to an action on a commitment when the speaker is the debtor, or to an action on a commitment content when the speaker is the debtor or the creditor. When an agent acts on the content of a commitment created by another agent, we refer to this as "*taking a position on a commitment content*".[3] However, it should be noted that the same utterance can lead both to take a position on the content of an existing commitment and to create a new commitment. Generally, a speech act leads to an action on a commitment and/or an action on a commitment content. Formally, in our framework a speech act can be defined using BNF notation as follows:

Definition 5.1 $SA(i_x, Ag_1, Ag_2, t_u, U) =_{def} Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$
$\mid Act\text{-}content(Ag_1, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))$
$\mid Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$ **&**
$Act\text{-}content(Ag_1, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))$

where $i, j \in \{1, 2\}$ and the meta-symbol "&" indicates "and".

This definition will be enriched when we establish the link between social commitments and arguments (Section 5.5)

[3] The term "*taking position*" is inspired by the work done by Rousseau, Moulin and Lapalme (1996) and extended by Bouzouba and Moulin (1998) and Bouzouba, Moulin and Kabbaj (2001). In these proposals, agents communicate by taking positions on the agents' private mental states which are exchanged by agents while conversing.

Let us take the previous example:
U1: *Quebec is the capital of Canada.*

The utterance *U1* leads to the creation of a new commitment:

$SA(I_0, Ag_1, Ag_2, t_{u1}, U1) =_{def}$
$\quad Create(Ag_1, t_{u1}, SC(Ag_1, Ag_2, t_{sc1}, (Capital(Canada, Québec), t_{\varphi 1})))$

U2: *No, the capital of Canada is Ottawa.*

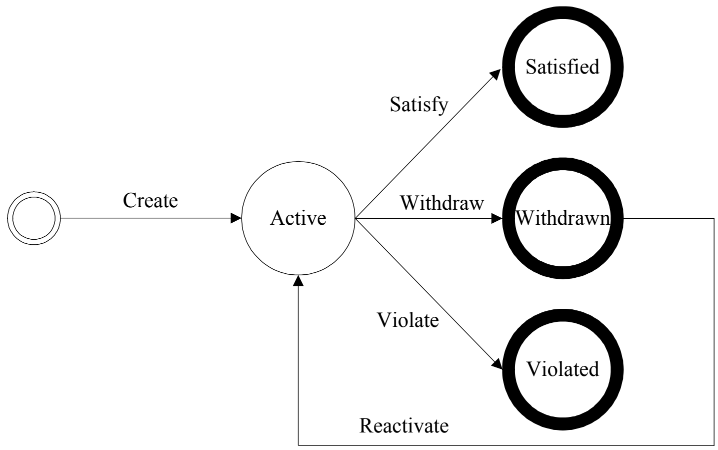
The utterance *U2* leads at the same time to a positioning on the content of the commitment created following the utterance *U1* and to the creation of another commitment. Formally:

$SA(I_1, Ag_2, Ag_1, t_{u2}, U2) =_{def}$
$\quad Refuse\text{-}content(Ag_2, t_{u2}, SC(Ag_1, Ag_2, t_{sc1}, (Capital(Canada, Quebec), t_{\varphi 1})))$
$\quad \& \ Create(Ag_2, t_{u2}, SC(Ag_2, Ag_1, t_{sc2}, (Capital(Canada, Ottawa), t_{\varphi 2})))$

5.3 The Notion of Commitment State

A commitment can evolve and be transformed as a result of the actions that the debtor performs on it (creation, withdrawal, reactivation, violation and satisfaction). Its content may also be transformed as a consequence of the actions that the debtor and the creditors apply to it (change, acceptance, justification, etc.). Therefore, agents act on their own commitments and on the contents of both their commitments and other agents' commitments. These actions lead to the transformation of these commitments and commitment contents. Hence, the notion of state makes it possible to capture the evolution of commitments and their contents. However, we must distinguish between the notion of the commitment state (Verdicchio and Colombetti, 2002) and the notion of the content state relative to this commitment as we propose here. Indeed, whenever an agent acts on its commitment, the commitment state is affected; whereas when an agent acts on the content of a commitment, the content state is transformed. Consequently, the notion of commitment state alone does not reflect the conversation dynamics since it only captures the debtor's actions on its commitment. The two states (the commitment state and the content state of the commitment) reflect this dynamics. This notion is of great importance since it allows us to keep a trace of the dialogue evolution in so far as each speech act leads to an action performed on a commitment or on its content. Contrary to the notion of the commitment store (Hamblin, 1970) which allows us only to track "who said what", the notion of state makes it possible to illustrate how participants change the dialogue state by performing actions on existing commitments or on their contents.

Here are the states that we propose to use in our model. Once created, a commitment will take the *active* state and its content takes the *submitted* state. This expresses the fact that the content is presented for possible negotiation. A commitment can be in one of four states: *active*, *satisfied*, *withdrawn*, and *violated* (Figure 5.5). A commitment content can be in one of nine states: *submitted*, *changed*, *refused*, *accepted*, *challenged*, *justified*, *contradicted*, *suspended*, *attacked* and *defended*. These states and the operations which trigger them depend on the commitment type. We notice that justification, contradiction, attack and defense are argumentation-related actions. This means that their semantics is defined using the argumentation notions (this aspect will be detailed in Chapter 7).



Fugure 5.5. Commitment state diagram

The set of different states of a commitment whose identifier is t_{sc} is denoted $S^{t_{sc}}$ and the set of different states of a commitment content whose identifier is t_{sc} is denoted $S^{t_{sc}}_{content}$. $S^{t_{sc}}$ and $S^{t_{sc}}_{content}$ are finite and ordered sets. The ordering relation $\prec$ between the elements of these sets is defined as follows:

Definition 5.2 $\forall s_1, s_2 \in S^{t_{sc}} (\in S^{t_{sc}}_{content}), s_1 \prec s_2$ iff the commitment (the commitment content) whose identifier is t_{sc} was in state s_1 before to be in state s_2.

The current state of a commitment (commitment content) whose identifier is t_{sc} is the biggest element of the set $S^{t_{sc}}$ ($S^{t_{sc}}_{content}$) according to the ordering relation $\prec$.

The following example illustrates this notion of state and its evolution:

U1: *The book is not allowed during the test.*
U2: *Why?*
U3: *Because the answers are given in this book.*
U4: *Ok, Thank you.*

By utterance *U1*, agent Ag_1 creates a commitment, whose state is “active”. The state of the content is “submitted”. Formally:

$SA(I_0, Ag_1, Ag_2, t_{u1}, U1) =_{def}$

$Create(Ag_1, t_{u1}, SC(Ag_1, Ag_2, t_{sc1}, (\neg Allow(Book, Test), t_{\varphi 1})))$

$S^{t_{sc1}} = \{\textbf{\textit{active}}\}$

$S^{t_{sc1}}_{content} = \{\textbf{\textit{submitted}}\}$

By utterance *U2*, agent Ag_2 challenges the content of the commitment identified by t_{sc1}. This commitment always remains in the "active" state, but its content takes the state "challenged". Formally:

$SA(I_1, Ag_2, Ag_1, t_{u2}, U2) =_{def}$

$Challenge\text{-}content(Ag_2, t_{u2}, SC(Ag_1, Ag_2, t_{sc1}, (\neg Allowed(Book, Test), t_{\varphi 1})))$

$S^{t_{sc1}} = \{\textbf{\textit{active}}\}$

$S^{t_{sc1}}_{content} = \{submitted, \textbf{\textit{challenged}}\}$ where $submitted \prec \textbf{\textit{challenged}}$

By utterance *U3*, agent Ag_1 creates a new commitment. The state of this commitment is "active", and the state of its content is "submitted". By the same utterance, this agent justifies the content of its commitment identified by t_{sc1}. The state of this commitment is always "active" and "justified" becomes the current state of its content. Formally:

$SA(I_2, Ag_1, Ag_2, t_{u3}, U3) =_{def}$

$Create(Ag_1, t_{u3}, SC(Ag_1, Ag_2, t_{sc2}, (Give(Answers, Book), t_{\varphi 2})))$

$\& \ Justify\text{-}content(Ag_1, t_{u3}, SC(Ag_1, Ag_2, t_{sc1}, (\neg Allow(Book, Test), t_{\varphi 1})))$

$S^{t_{sc1}} = \{\textbf{\textit{active}}\}$

$S^{t_{sc1}}_{content} = \{submitted, challenged, \textbf{\textit{justified}}\}$ where $challenged \prec \textbf{\textit{justified}}$

$S^{t_{sc2}} = \{\textbf{\textit{active}}\}$

$S^{t_{sc2}}_{content} = \{\textbf{\textit{submitted}}\}$

By utterance *U4*, agent Ag_2 accepts the content of the commitment identified by t_{sc2}. Thus, "satisfied" becomes the current state of this commitment and "accepted" becomes the state of its content. Consequently, this agent also accepts the content of the commitment identified by t_{sc1}. Thus, "satisfied" and "accepted" are the current states respectively of this commitment and its content. Formally:

$SA(I_3, Ag_2, Ag_1, t_{u4}, U4) =_{def}$

$Accept\text{-}content(Ag_2, t_{u4}, SC(Ag_1, Ag_2, t_{sc2}, (Give(Answers, Book), t_{\varphi 2})))$

$\& \ Accept\text{-}content(Ag_2, t_{u4}, SC(Ag_1, Ag_2, t_{sc1}, (\neg Allow(Book, Test), {}_{\varphi 1})))$

$S^{t_{sc1}} = \{active, \textbf{\textit{satisfied}}\}$ where $active \prec \textbf{\textit{satisfied}}$

$S^{t_{sc1}}_{content} = \{submitted, challenged, justified, \textbf{\textit{accepted}}\}$ where $justified \prec \textbf{\textit{accepted}}$

$S^{t_{sc2}} = \{active, \textbf{\textit{satisfied}}\}$ *where* $active \prec \textbf{\textit{satisfied}}$

$S^{t_{sc2}}_{content} = \{submitted, \textbf{\textit{accepted}}\}$ *where* $submitted \prec \textbf{\textit{accepted}}$

5.4 Taxonomy of Commitment Types

In the literature (Walton and Krabbe, 1995), (Singh, 1999), (Fornara and Colombetti, 2002), several commitment types have been proposed. In our approach we distinguish *absolute commitments*, *conditional commitments* and *commitment attempts*.

5.4.1 Absolute Commitments

Absolute commitments are commitments whose fulfillment does not depend on any particular condition. An absolute commitment is denoted:

$ABC(Ag_1, Ag_2, t_{abc}, (\lambda, t_\lambda)))$

Two types can be distinguished: *propositional commitments* and *action commitments*.

A. Propositional Commitments

Propositional commitments are related to the state of the world. They are expressed by assertives or by speech acts of declaratory and expressive types. They can be directed towards the past, the present, or the future. We denote a propositional commitment as follows:

$PC(Ag_1, Ag_2, t_{pc}, (p, t_p))$
where p is the proposition on which Ag_1 commits.

Example:
U: *The door is open*

$SA(I_0, Ag_1, Ag_2, t_u, U) =_{def} Create(Ag_1, t_u, PC(Ag_1, Ag_2, t_{pc}, (open(door), t_p)))$
such that $t_{pc} = t_p$.

Because propositional commitments are particular cases of social commitments, the relationship between t_u and t_{pc} is similar to the one existing between t_u and t_{sc}.

B. Action Commitments

Action commitments (also called *commitments to a course of action*) are directed towards the present or the future and are related to actions that the debtor is committed to perform. The fulfillment and the violation of such commitments depend on the performance of the underlying action and the specified delay. This type of commitment is typically conveyed by promises. We denote an action commitment as follows:

$AC(Ag_1, Ag_2, t_{ac}, ((\alpha, p), t_\alpha))$

where α is the action to be performed, and by performing α, the proposition p becomes true. The relationship between the symbol action α and the proposition p is similar to the relationship existing between α and p in the operator $<\alpha>p$ of dynamic logic. Adding a proposition to the notation of an action commitment enables us to define the semantics of this commitment using this operator. This aspect will be detailed in Chapter 7. We notice here that p does not need a temporal argument because if α is performed at t_α, then p becomes true at this moment.

Example:
U: *I will give you 10 dollars in one hour*

$SA(I_0, Ag_1, Ag_2, t_u, U) =_{def}$
$\quad Create(Ag_1, t_u, AC(Ag_1, Ag_2, [t_u, t_u+1h], ((\alpha, Give(Ag_1, Ag_2, 10\ dollars)), t_\alpha)))$
where α is an action symbol whose performance makes the proposition $Give(Ag_1, Ag_2, 10\ dollars))$ true, and $t_\alpha = t_u + 1h$.

The state diagram of an absolute commitment is similar to that of Figure 5.5. Figure 5.6 presents the state diagram associated with the content of such a commitment. It contains the possible states for the commitment contents and the transitions corresponding to the operations, which can be applied to these contents. The dotted transitions in the figure correspond to the creditor's actions and the non-dotted transitions correspond to the debtor's actions. These operations are reflected by the participants' utterances. Thus, the debtor can submit a commitment content, contradict it, justify it, defend it and change it. The creditor can accept this content, refuse it, challenge it and attack it.

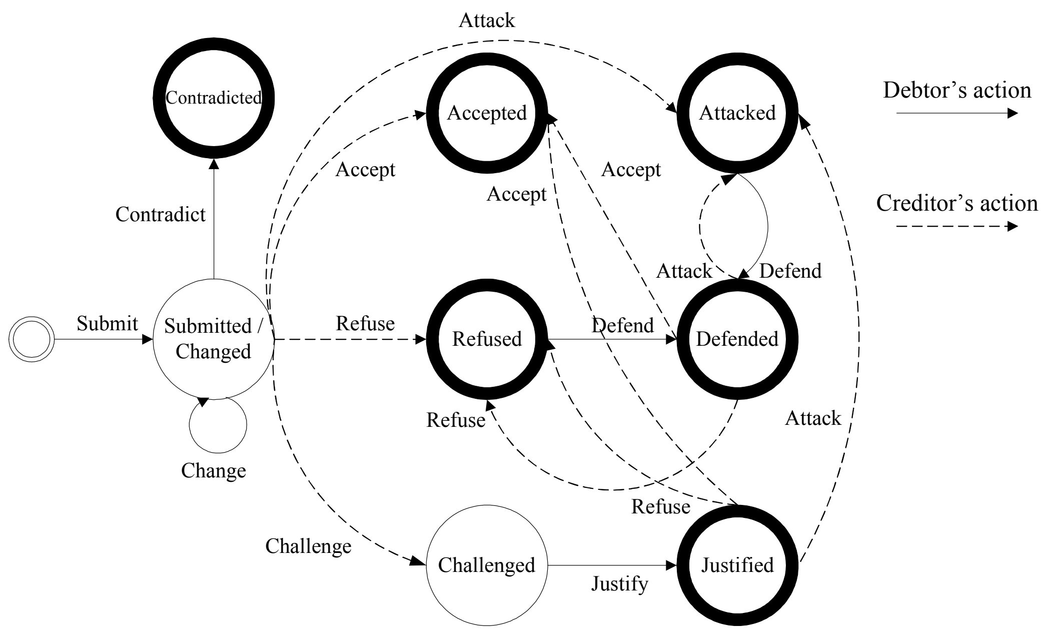


Figure 5.6. State diagram associated to the content of an absolute commitment

A commitment towards the present or the future can be interpreted either as a propositional commitment or as an action commitment. For example, the utterance *"tomorrow the door will be open"* may be interpreted as a propositional commitment made by the speaker on a future state of the world. It can also be interpreted as an action commitment if the speaker is responsible for opening the door in question. Therefore, the commitment made by the speaker depends on the conversation context. It is in this sense that social context is a fundamental issue in communication (Moulin, 1998). In particular, this allows us to handle properly indirect speech acts (Bouzouba and Moulin, 1999).

In our framework, there is no explicit relation between propositional commitments and action commitments. When the current state of the world does not satisfy a propositional commitment, we speak about a violation of this commitment. There is no rule indicating that an agent develops an action commitment to make the content of its propositional commitment true when this commitment becomes violated. A propositional commitment is a commitment about a state of the world that the debtor agent can or cannot realize. In contrast, an action commitment is a commitment about an action that the debtor commits to perform in the present or in the future.

5.4.2 Conditional Commitments

Absolute commitments do not consider the conditions that may restrain their fulfillment. However, in several cases, agents need to make commitments not in absolute terms but under given conditions. Another commitment type is therefore required in order to be able to capture situations defined by certain conditions. These commitments are said to be *conditional*. The structure of a *conditional commitment*, which must reflect the underlying condition, is different from the structure of a social commitment. We denote a conditional commitment as follows:

$$CC(Ag_1, Ag_2, t_{cc}, ((p, t_p), (\lambda, t_\lambda)))$$

This commitment expresses the fact that if p is true at time t_p, then Ag_1 will be committed towards Ag_2 to perform λ or so that λ is true at time t_λ. The future for a conditional commitment depends not only on time but also on the satisfaction of the underlying condition. Like for absolute commitments, we can distinguish between conditional commitments about propositions denoted:

$$PCC(Ag_1, Ag_2, t_{pcc}, ((p, t_p), (p', t_{p'})))$$

and conditional commitments about actions denoted:

$$ACC(Ag_1, Ag_2, t_{acc}, ((p, t_p), ((\alpha, p'), t_\alpha)))$$

The relationship between the action symbol α and the proposition p' is similar to the one existing between this action and this proposition in action commitments.

This distinction is implicit since according to the axiom ($A1$) a conditional commitment becomes an absolute commitment when the condition is satisfied.

$$A1: CC(Ag_1, Ag_2, t_{cc}, ((p, t_p), (\lambda, t_\lambda))) \wedge (p, t_p) \equiv true$$
$$\Rightarrow ABC(Ag_1, Ag_2, t_{abc}, (\lambda, t_\lambda))$$

where $t_{abc} = t_{cc}$.

The state diagram associated with a conditional commitment is similar to that of Figure 5.5. The state diagram associated with the content of such a commitment is identical to that of the content of an absolute commitment (Figure 5.6). Indeed, we can consider any social commitment as a conditional commitment whose underlying condition is always true. Thus, we have the following (syntactical) equivalence:

$$SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)) \equiv \forall t\ CC(Ag_1, Ag_2, t_{sc}, ((true, t), (\varphi, t_\varphi)))$$

Example:
U: *If industrial countries ratify the Kyoto Protocol, it can take effect*

$SA(I_0, Ag_1, Ag_2, t_u, U) =_{def}$
$Create(Ag_1, t_u, PCC(Ag_1, Ag_2, t_{cc},$
$((ratify(industrial\ countries, Kyoto\ Protocol), t_p), (can\text{-}take\text{-}effect(Kyoto\ Protocol), t_{p'})))$

where $t_{p'} = t_p$.

5.4.3 Commitment attempts

The commitments described so far directly concern the debtor who commits either that a certain fact is true or to perform certain action. For example, these commitments do not allow us to explain the fact that an agent asks another one to be committed to perform an action (by a speech act of a directive type). To solve this problem, we propose the concept of *commitment attempt* inspired by the notion of *pre-commitment* proposed in (Colombetti, 2000). We consider a commitment attempt as a request made by a debtor to push a creditor to be committed. Thus, when an agent Ag_1 requests another agent Ag_2 to do something, we say that the first agent is trying to induce the other agent to make a commitment. In this chapter, we denote a commitment attempt as follows:

$$CT(Ag_1, Ag_2, t_{ct}, (\varphi, t_\varphi))$$

where φ is the content of the commitment attempt. This formulation seems more intuitive than Colombetti's one according to which the agent Ag_2 is the debtor and the agent Ag_1 is the creditor. In Chapter 7, we will improve this notation in order to be able to express the semantics of this type of commitments using an existential qualifier.

A commitment attempt about a proposition p is denoted:

$PCT(Ag_1, Ag_2, t_{pct}, (p, t_p))$

A commitment attempt about an action α whose performance makes true a proposition p is denoted:

$ACT(Ag_1, Ag_2, t_{act}, ((\alpha, p), t_\alpha))$

The relationship between the action symbol α and the proposition p is similar to the one existing between α and p in action commitments.

Example:
U: *Could you call me at 4PM?*

$SA(I_0, Ag_1, Ag_2, t_u, U) =_{def}$
$Create(Ag_1, t_u, ACT(Ag_1, Ag_2, t_{act}, ((\alpha, call(Ag_2, Ag_1)), 4PM)))$

A commitment attempt is thought of as a type of social commitment because it conveys content which is made public once the attempt is performed. However, in our approach, there is a true commitment only after the creditor agent reacts in response to the commitment attempt by accepting it or by refusing it. We speak here about the "*co-construction*" of social commitments by the two interlocutors. This idea is similar to the one proposed by Rousseau, Moulin and Lapalme (1996) in which agents co-construct speech acts using their private mental states. The debtor and the creditor of a commitment attempt can act both on the attempt and on its content. On the one hand, the creditor agent reserves the right to accept a commitment attempt, to refuse it or to suspend it (for example by asking for a period of time for thought). It can also challenge the content of a commitment attempt. On the other hand, the debtor agent can withdraw a commitment attempt. It can also change the content of a commitment attempt and justify it. The states of a commitment attempt and those of its content can also be described by a state diagram. Figure 5.7 illustrates the state diagram associated to the content of a commitment attempt. Like a social commitment, a commitment attempt can be absolute (*ABCT*) or conditional (*CCT*). An absolute commitment attempt is denoted:

$ABCT(Ag_1, Ag_2, t_{abct}, (\gamma, t_\gamma))$

A conditional commitment attempt is used for example when an agent asks another one to do some thing if a certain condition is true. A conditional commitment attempt about a proposition is denoted:

$CCTP(Ag_1, Ag_2, t_{cctp}, ((p, t_p), (p', t_{p'})))$

A conditional commitment attempt about an action is denoted:

$CCTA(Ag_1, Ag_2, t_{ccta}, ((p, t_p), ((\alpha, p'), t_\alpha)))$

where p is the underlying condition.

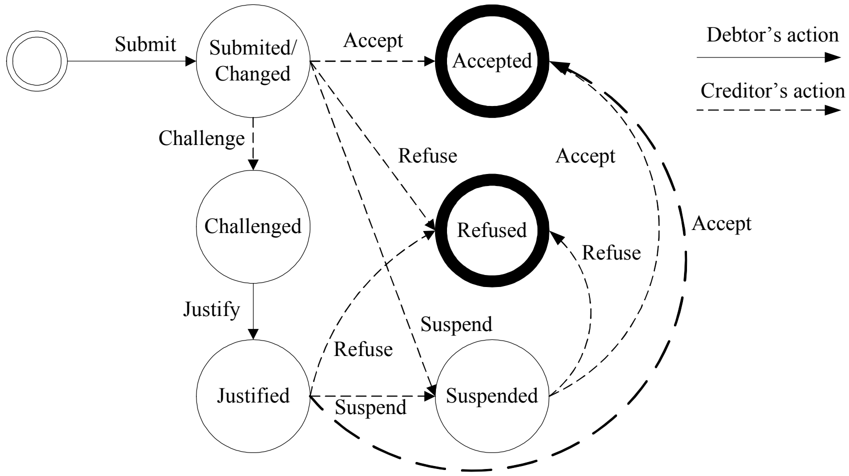


Figure 5.7. State diagram associated to the content of a commitment attempt

The refusal and the acceptance of a commitment attempt automatically lead to the creation of a new commitment that is in the active state. The two following rules illustrate this characteristic when the commitment attempt relates to a proposition or an action γ:

$R1$: $Create(Ag_1, t_u, ABCT(Ag_1, Ag_2, t_{abct}, (\gamma, t_\gamma)))$
$\quad \& \ Refuse\text{-}content(Ag_2, t_{refuse}, ABCT(Ag_1, Ag_2, t_{abct}, (\gamma, t_\gamma)))$
$\quad \Rightarrow Create(Ag_2, t_{refuse}, ABC(Ag_2, Ag_1, t_{abc}, (\neg\gamma, t_\gamma)))$

Syntactically, if λ is an action, $\neg\lambda$ indicates that this action will not be performed.

$R2$: $Create(Ag_1, t_u, ABCT(Ag_1, Ag_2, t_{abct}, (\gamma, t_\gamma)))$
$\quad \& \ Accept\text{-}content(Ag_2, t_{accept}, CT(Ag_1, Ag_2, t_{abct}, (\gamma, t_\gamma)))$
$\quad \Rightarrow Create(Ag_2, t_{accept}, ABC(Ag_2, Ag_1, t_{abc}, (\gamma, t_\gamma)))$

When the commitment attempt relates to a condition, the rules $R1$ and $R2$ become:

$R1'$: $Create(Ag_1, t_u, CCT(Ag_1, Ag_2, t_{cct}, ((p, t_p), (\gamma, t_\gamma))))$
$\quad \& \ Refuse\text{-}content(Ag_2, t_{refuse}, CCT(Ag_1, Ag_2, t_{cct}, ((p, t_p), (\gamma, t_\gamma))))$
$\quad \Rightarrow Create(Ag_2, t_{refuse}, SC(Ag_2, Ag_1, t_{sc}, (\neg((p, t_p), (\gamma, t_\gamma)))))$

$R2'$: $Create(Ag_1, t_u, CCT(Ag_1, Ag_2, t_{cct}, ((p, t_p), (\gamma, t_\gamma))))$
$\quad \& \ Accept\text{-}content(Ag_2, t_{accept}, CCT(Ag_1, Ag_2, t_{cct}, ((p, t_p), (\gamma, t_\gamma))))$
$\quad \Rightarrow Create(Ag_2, t_{accept}, CC(Ag_2, Ag_1, t_{cc}, ((p, t_p), (\gamma, t_\gamma))))$

According to rule $R1'$, refusing a commitment attempt which relates to a condition consists of refusing its content without committing towards its condition. However, according to rule $R2'$, accepting a commitment attempt consists of accepting it under its condition, which leads to a conditional commitment.

5.5 The Link between Argumentation and Commitments

Argumentation is based on the construction of arguments and counter-arguments (arguments attacking other arguments), the comparison of these various arguments and finally the selection of the arguments that are considered to be acceptable. A defeasible argumentation system essentially includes a logical language *L*, a definition of the argument concept, a definition of the attack relation between arguments and finally a definition of acceptability. In our model the formal definitions of these notions are inspired by (Elvang-Goransson et al., 1993). Here Γ indicates a knowledge base, $\vdash$ stands for classical inference and $\equiv$ for logical equivalence.

Definition 5.3 *An argument is a pair* (H, h) *where h is a formula of L and H a sub-set of* Γ *such that: i*) *H is consistent, ii*) $H \vdash h$ *and iii*) *H is minimal, so that no subset of H satisfying both i and ii exists. H is called the support of the argument and h its conclusion.*

Definition 5.4 *Let* (H_1, h_1), (H_2, h_2) *be two arguments.*
(H_1, h_1) *attacks* (H_2, h_2) *iff* $H_2 \vdash \neg h_1$. *In other words, an argument is attacked if and only if there exists an argument for the negation of its conclusion.*

The concept of acceptability is defined as follows (Dung, 1995):

Definition 5.5 *An argument* (H, h) *is acceptable for a set S of arguments iff for any argument* (H', h'): *if* (H', h') *attacks* (H, h) *then* (H', h') *is attacked by S.*

Intuitively, an argument is acceptable if it is not attacked, if it defends itself against all its attackers, or if it is defended by an acceptable argument.

According to (Dung, 1995), any argumentation system includes two essential elements: one element is used to build and generate arguments, the other is used to analyze these arguments by determining their acceptability. This view is important for our communication model. Indeed, agents must reason about their own mental states in order to build arguments in favor of their future commitments, as well as about other agents' commitments in order to be able to take position with regard to the contents of these commitments. Surely, an argumentation system is essential to help agents to act on commitments and on their contents. However, reasoning about other social attitudes should be taken into account in order to explain agents' decisions. This aspect will be discussed in Chapter 9, in which we highlight the importance of agents' trustworthiness to decide, in some cases, about the acceptance of arguments.

The systems proposed in the literature, for example in (Dung, 1995), (Vreeswijk, 1997), (Amgoud, 1999) do not discuss how arguments can support communicative actions. We will specify this here. In fact, before committing to some proposition *h* being true (i.e. before creating a commitment whose content is *h*, the speaker agent must use its argumentation system to build an argument (H, h). On the other side, the addressee agent must use its own argumentation system to select the answer it will give (i.e. to select the appropriate manipulation of the content of an existing commitment). For example, an agent Ag_1 accepts the commitment content h proposed by another agent Ag_2 if Ag_1 is able to build an argument which supports this content from its knowledge base which is assumed to be consistent. If Ag_1 has an argument $(H', \neg h)$, then it refuses or attacks this commitment content. If Ag_1 does not have any argument for h, or for $\neg h$, then it must ask for an explanation. In this case, Ag_2 must justify the content h.

Thus, an agent should always use its argumentation system before creating a new commitment or positioning itself on an existing commitment and on its content. Consequently, an argument of an agent Ag_1 must support an action of this agent on a given commitment and/or on its content. Formally, an agent Ag_k's argument supporting its action at time t_u on a given commitment is denoted:

$$Arg(Ag_k, H, Act(Ag_k, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi))))$$

An Ag_k's argument supporting its action at time t_u on a given commitment content is denoted:

$$Arg(Ag_k, H, Act\text{-}content(Ag_k, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi))))$$

with H being the support of the argument and the agent identifiers i, j and k verify:
$i, j, k \in \{1, 2\}$ and $i \neq j$.

In the first formula, H is the support of the action *Act* performed by agent Ag_k on the commitment identified by t_{sc}. In the second formula, H is the support of the action *Act-content* performed by agent Ag_k on the content of this commitment. We notice that this support holds at the moment of the action. Thus, according to the nonmonotonicity of arguments, it is possible that this support becomes invalid if new information becomes available for Ag_k. In this case, Ag_k must update its knowledge base by removing the invalid argument and adding the new valid argument.

We notice that there is a logical relation between arguments supporting actions and arguments supporting propositions. The argument supporting an action is the argument supporting the proposition that becomes true when the action is performed. This relation is similar to the relation existing between actions and propositions in a dynamic logic (Harel, 1984). In this logic, the semantics of an action is defined as follows:

$$\mathcal{M} \models_w \langle\alpha\rangle p \textit{ iff } (\exists w' : R_\alpha(w, w') \,\&\, \mathcal{M} \models_{w'} p)$$

This means that in a Kripke structure $\mathcal{M}$ (the model) the action α is satisfied in a world w iff there is an R_α – accessible world w' in which the proposition p becomes true (R_α is called *accessibility relation*). The idea is that by doing the action α the proposition p becomes true in an accessible world. In our approach an argument H supporting an action *Act* (respectively an action *Act-content*) performed on a commitment whose content is φ (respectively on the content φ) is satisfied in a world w iff there is an R_{Act} – accessible world w' in which H supports φ or $\neg\varphi$. R_{Act} is the accessibility relation associated with the action *Act* (respectively *Act-content*). For example, the argument supporting an acceptance action of a social commitment content is the argument supporting this content.

In fact, the relation between H and the commitment content φ depends on the values of *Act* and *Act-content*. Thus, for an absolute or a conditional commitment we have the following axiom:

A2: $Act \in \{Create, Satisfy\} \Rightarrow H \vdash \varphi$

$Act = Withdraw \Rightarrow \nexists H: H \vdash \varphi$

$Act\text{-}content \in \{Submit\text{-}content, Accept\text{-}content, Change\text{-}content, Justify\text{-}content, Defend\text{-}content\} \Rightarrow H \vdash \varphi$

$Act\text{-}content \in \{Refuse\text{-}content, Attack\text{-}content\} \Rightarrow H \vdash \neg\varphi$

For example, the first rule indicates that if *Act* takes the value "*Create*" or "*Satisfy*", then H supports φ.

To illustrate this idea, let us take the following example between agents Ag_1 and Ag_2 that we dealt with in Section 5.3:

U1: *The book is not allowed during the test.*
U2: *Why?*
U3: *Because the answers are given in this book.*
U4: *Ok, Thank you.*

We suppose that the Ag_1's knowledge base contains the arguments (H, φ) and (φ, φ), and the Ag_2's knowledge base contains the argument (φ, φ) where $H = Give(Answers, Book)$ and $\varphi = \neg Allow(Book, Test)$. By utterance *U3*, agent Ag_1 presents the support H in order to justify the content φ of the commitment identified by t_{sc1}. Formally we have:

$$Arg(Ag_1, Give(Answers, Book), Create(Ag_1, t_{u1}, SC(Ag_1, Ag_2, t_{sc1}, (\neg Allow(Book, Test), t_{\varphi 1}))))$$

For a commitment attempt we have the following axiom:

A3: $Act = Create \Rightarrow \nexists H: H \vdash \varphi \text{ or } H \vdash \neg\varphi$

$Act\text{-}content = Suspend\text{-}content \Rightarrow \nexists H: H \vdash \varphi \text{ or } H \vdash \neg\varphi$

$Act\text{-}content = Accept\text{-}content \Rightarrow H \vdash \varphi$

$Act\text{-}content = Refuse\text{-}content \Rightarrow H \vdash \neg\varphi$

$Act\text{-}content \in \{Change\text{-}content, Justify\text{-}content\} \Rightarrow H \vdash \varphi$

An agent can create a commitment attempt related to a proposition p, if it does not have any argument for p or for $\neg p$. This reasoning is also valid for a commitment attempt related to a condition or an action. In the case of an action α the agent does not have any argument for or against the proposition p that becomes true by performing α. We notice here that this aspect cannot be verified because the fact that an agent does not have an argument for or against a proposition is related to its internal state. The idea is that an agent can create a commitment attempt related to a proposition p even if it has an argument for p. In addition, the creation of a commitment attempt related to an action can also depend on the context. For example, to create a commitment attempt in the form of an order, the debtor must have the social capacity to give orders to the other agent.

When considering the creation of a new commitment, the agent must also have a reason supporting it (a kind of goal to be achieved). In our approach, this reason is considered as an argument for the action, which is different from the argument that supports the content. Let us take the following example:

Example:
U: *The book "Agent Technology" is interesting*
$SA(i_0, Ag_1, Ag_2, t_u, U) =_{def}$
$Create(Ag_1, t_u, PC(Ag_1, Ag_2, t_{pc}, (Interesting(book\ Agent\ Technology), t_p)))$
such that $t_{pc} = t_p$.

To create this commitment, Ag_1 must have a *reason* to do it, as for example in order to ask Ag_2 to buy the book. This reason can be considered as an argument supporting the creation action that is different from the argument supporting the content, corresponding for example to "this book is interesting because its editors are well known authors". It is thus significant in this case to distinguish the argument supporting the creation action itself and the argument supporting the content. Generally, the arguments supporting the creation actions are not expressed in speech acts, but correspond to agents' private goals. We define an argument supporting a creation action as follows:

Definition 5.6 *An argument supporting a creation action of a commitment is a pair* $(\beta, Create(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi))))$ *such that* i) β *is a proposition representing an agent* Ag_1*'s goal and ii*) $\varphi \vdash \beta$.

The proposition β can be any Ag_1's goal. For example, the goal can be just to inform another agent that some thing is true.

After the introduction of the argumentation in our approach, we note that a speech act can lead to an action not only on a commitment as explained in Section 5.2, but also on an argument. An agent can thus accept, refuse, defend or attack an argument. Thus, using BNF notation, we have the following definition improving Definition 5.1:

Definition 5.7 $SA(I_x, Ag_1, Ag_2, t_u, U) =_{def}$
$Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$

$\mid Act\text{-}content(Ag_1, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))$

$\mid Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$

$\& \ Act\text{-}content(Ag_1, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))$

$| \ Act\text{-}content(Ag_1, t_u, Arg(Ag_1, H, Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))))$

$| \ Act\text{-}content(Ag_1, t_u, Arg(Ag_k, H, Act\text{-}content(Ag_k, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))))$

$| \ Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$
$\& \ Act\text{-}content(Ag_1, t_u, Arg(Ag_1, H, Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))))$

$| \ Act(Ag_1, t_u, SC(Ag_1, Ag_2, t_{sc}, (\varphi, t_\varphi)))$
$\& \ Act\text{-}content(Ag_1, t_u, Arg(Ag_k, H, Act\text{-}content(Ag_k, t_u, SC(Ag_i, Ag_j, t_{sc}, (\varphi, t_\varphi)))))$

where $i, j, k, \in \{1, 2\}$ and $i \neq j$.

In Chapter 6 (Section 6.3), we will give a detail example illustrating this definition.

5.6 Communication Model

In the previous sections we proposed a formulation of the pragmatics of agent communication using the notion of actions that agents perform on commitments and the arguments enabling agents to select the communicative act to be performed. In this section, we propose an architecture of a communication model in which this approach takes place. In fact, this model combines the three approaches discussed in our taxonomy of the prior approaches (see Chapter 4). It is based on a hybrid approach that we call MSA (Mental-Social-Argumentative). Indeed, if they are taken individually, these three approaches do not allow us to model all the aspects of agent communication. For this reason, we suggest to combine them in a unified approach. In addition, conversation is a cognitive and social activity, which requires a mechanism making it possible to reason about mental states, about what other agents say (public aspects), and about the social aspects (conventions, standards, obligations, etc). These three approaches are thus not exclusive but rather complementary.

The MSA approach has the advantage of capturing simultaneously the mental aspect that characterizes agents participating in a conversation, the social aspect that reflects the context in which these agents communicate, and the reasoning aspect which is essential to be able to take part in conversations. The combination of commitments and arguments seems essential to us because agents must be able to justify the claims to which they are committed and to justify their actions on commitments. This justification cannot be made if agents do not have the necessary argumentation mechanisms. In addition, the combination of commitments and private mental states is necessary because public commitments reflect these mental states that contain additional information motivating the agent's communicative acts. Finally, the combination of argumentation and mental states is significant because agents have to reason about their mental states before committing in a conversation.

The communication model is composed of three layers: the conversation layer, the commitment/argument layer and the cognitive layer (see Figure 5.8). The abstraction levels justify this stratification in layers. The conversation layer is directly observable because it is composed of the speech acts that agents perform. These acts are not performed in an isolated way, but within a particular conversation. The commitment/argument layer is used to correctly manage the social commitments and the arguments that are related to the conversation. Finally, the cognitive layer is used to take into account the private mental states of agents, the social relations and other elements that agents use in order to communicate.

In order to allow conversational agents to suitably use the communication model, this model must be compatible with the agent architecture. Thus, we propose an architecture of conversational agents, which is composed of three models: the mental model, the social model and the reasoning model (Figure 5.8). The mental model includes beliefs, desires, goals, etc. The social model captures the social concepts such as conventions, roles, etc. Social commitments constitute a significant component of this model. The commitments that an agent makes public when performing speech acts are different from the private mental states, but these two elements are not independent. Indeed, social commitments reflect mental states. Thus, agents must use their reasoning capabilities to reason about their mental states before producing or manipulating social commitments. The agent’s reasoning capabilities are represented by the reasoning model using an argumentation system. The conversational agent model also involves by general knowledge, such as the knowledge of the conversation subject. An agent will use this knowledge in order to build the common ground that it must share with its partners. The notion of common ground introduced by the philosophers of language Clark and Haviland (1974) indicates the set of knowledge, beliefs, and presuppositions, which agents believe that they share during their conversations.

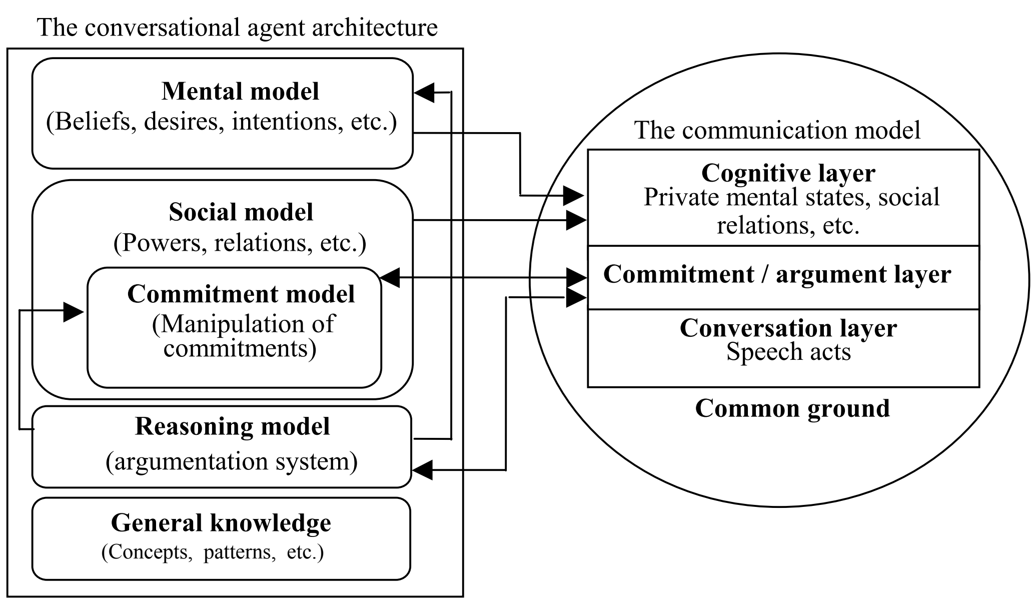


Figure 5.8. The links between the conversational agent architecture and the communication model

5.7 Discussion

In this chapter, we argued that the three approaches discussed in Chapter 4 can be successfully combined in one pragmatic approach. This unified approach has the advantage of capturing the external public aspect of agent communication and the internal private aspect of agents. The main idea of this approach is that agent communication is considered as actions that agents perform on social commitments and arguments. The dynamics of agent conversation is represented by this notion of actions. In addition, the notion of commitment state enables us to reflect the evolution of agent conversations. The current state of a conversation is clearly described by the state of the different commitments. The use of argumentation allows agents to participate in complete conversations because at each moment they can select the next action to be performed. This approach can be used to specify protocols that are more flexible than classical protocols in the sense that participating agents can make decisions by reasoning about the current state of their conversations. Because it captures the private and the public aspects of agent communication, this approach can also be used to specify the different dialogue types according to the classification proposed by Walton and Krabbe (1995) (see Chapter 2). Thus, in Chapter 9, we will show how it can be used to specify dialogue games in the case of a persuasion dialogue game protocol. We also used this approach to develop a computational model for the dialogization and the implicit information in a communicational model (Bouzouba et al., 2004). A comparison between our approach and other proposals will be made in the next chapter.

Chapter 6*

Commitment and Argument Network

In this chapter, we propose a formal framework called Commitment and Argument Network (CAN) which offers an external representation of conversations between agents. This framework is based on our pragmatic approach proposed in the previous chapter. Using this formalism allows us: (1) *to represent the dynamics of conversations between agents;* (2) *to analyze agent conversations;* (3) *to help autonomous agents take part in conversations.*

6.1 Introduction

As outlined in Chapters 3 and 4, several proposals on agent communication have been focused on modeling pragmatic and semantic issues. However, few researchers have addressed the issue of representing the dynamics of conversations. The purpose of this chapter is to propose a formal framework called *Commitment and Argument Network* (CAN) for representing these dynamics. This framework represents agent actions likely to take place in a conversation. As outlined in Chapter 5, these actions are interpreted in terms of the creation of and positioning on social commitments and arguments. The proposed formalism allows us to model the dynamics of conversations and offers an external representation of the conversational activity. An external representation of a conversation is a representation of the different communicative acts that can be observed by an external observant. This notion of external representation (Clark, 1996) is extremely useful because it provides conversational agents with a common understanding of the current state of the conversation and its evolution (Rousseau et al., 1996). Based on our formalism, a model is made available to agents which they can access simultaneously. This formalism clearly illustrates the creation steps of new commitments and the positioning steps on these commitments, as well as the argumentation steps.

In the previous chapter, we presented our formulation of commitments and of the relations between these commitments and arguments. Indeed, our goal is to develop a pragmatic approach based on commitments and arguments. This approach aims at providing software agents with a flexible means to interact. Thus, agents can participate in conversations by

* We would like to thank John-Jules Ch. Meyer, Frank Dignum from Utrecht University, Intelligent Systems Group, Iyad Rahwan from University of Melbourne, and Yves Lespérance from York University for the helpful discussions about the formalism presented in this chapter. This formalism is published in (Bentahar et al., 2004b, 2004c).

manipulating commitments and by producing arguments. It is the agents' responsibility (and not the designers' role) to choose, in an autonomous way, the actions to be performed by using their argumentation systems. In this chapter, we show how a conversation can be modeled using the CAN formalism on the basis of this approach. In a conversational activity, agents manage commitments and arguments. Our purpose is to represent the dynamics of conversations using this formalism. This representation allows us to ensure conversational consistency and coherence in terms of the actions performed by agents on the commitments and arguments. Indeed, this framework has two objectives: it can be used to analyze conversations, as well as provide a means for allowing agents to take part in conversations.

The rest of this chapter is structured as follows. In Section 6.2, we present the foundations of the CAN formalism. In Section 6.3, we give an example illustrating how an agent conversation can be represented and analyzed using this framework. In Section 6.4, we demonstrate how our formalism can be used as a means permitting agents to take part in conversations. Two additional examples using additional commitment types are then presented in Section 6.5. We show, in Section 6.6, that the CAN framework can represent any argumentative conversation. Finally in Section 6.7, we compare our pragmatic approach and our framework to related work.

6.2 Formal Definition

In this chapter, we simplify the notation of a social commitment by omitting the argument related to content time. A social commitment will be denoted: $SC(Ag_1, Ag_2, t, \varphi)$ instead of $SC(Ag_1, Ag_2, t_{sc}, \varphi, t_\varphi)$.

A commitment and argument network is a mathematical structure which we define formally as follows (the explanation of the different components will be given after):

Definition 6.1 *A commitment and argument network is a 12-uple*:
$<A, E, SC(Ag_1, Ag_2, t_0, \varphi_0), T, \Omega, \Sigma, F_{E\Sigma}, F_{E\Sigma\Sigma}, F_\Omega, F_{A\Sigma\Omega}, F_{A\Omega\Omega}, F_{E\Omega\Sigma}>$
where:

- A: *a finite and nonempty set of participants.*

In this chapter, we suppose that: $A = \{Ag_1, Ag_2\}$.

- E: *a finite and nonempty set of social commitments.*

These commitments can be absolute commitments (*ABC*), *conditional commitments* (*CC*) *or commitment attempts* (*CT*) .
$E=\{SC(Ag_1, Ag_2, t_0, \varphi_0), \ldots, SC(Ag_i, Ag_j, t_n, \varphi_n)\}$ *such that*: $i, j \in \{1, 2\}$.

- $SC(Ag_1, Ag_2, t_0, \varphi_0)$: *a distinguished element of E indicating the initial commitment.*

This element allows us to define the subject of a conversation.

- T: *the set of time points.*

$T = \{t_0, \ldots, t_n\}$.

- Ω: *the set of creation and positioning actions.*

Ω = {*Create*, *Withdraw*, *Reactivate*, *Satisfy*, *Violate*, *Accept-content*, *Refuse-content*, *Challenge-content*, *Suspend-content*, *Change-content*}.

- *Σ: the set of argumentation relations.*

Σ = {Defend-content, Attack-content, Justify-content, Contradict-content}.

- *$F_{E\Sigma}$: a partial function relating one commitment to a second commitment using one argumentation relation and a time unit. We call this function: commitment-argument-commitment function.*

$F_{E\Sigma}: E \times E \mapsto \Sigma \times T$

- *$F_{E\Sigma\Sigma}$: a partial function relating one commitment to a pair made up of an argumentation relation and a time point using one argumentation relation and another time point. We call this function commitment-argument-argument function.*

$F_{E\Sigma\Sigma}: E \times \Sigma \times T \mapsto \Sigma \times T$

- *F_{Ω}: a partial function relating an agent (a participant) to a commitment using a set of pairs made up of a creation or a positioning action and a time point. We call this function agent-commitment function.*

$F_{\Omega}: A \times E \mapsto 2^{\Omega \times T}$

- *$F_{A\Sigma\Omega}$: a partial function relating an agent to an argumentation relation characterized by a time point using a set of pairs made up of a creation or positioning action and a time point. We call this function agent-action-argument function.*

$F_{A\Sigma\Omega}: A \times \Sigma \times T \mapsto 2^{\Omega - \{Change\text{-}content\} \times T}$

- *$F_{A\Omega\Omega}$: a partial function relating an agent to a creation or a positioning action characterized by a time unit using a set of pairs made up of a positioning action and a time unit. We call this function agent-action-action function.*

$F_{A\Omega\Omega}: A \times \Omega \times T \mapsto 2^{\Omega - \{Create,\ Satisfy,\ Violate,\ Change\text{-}content\} \times T}$

- *$F_{E\Omega\Sigma}$*: a partial function relating a commitment to a creation or a positioning action characterized by a time unit using one argumentation relation. *We call this function commitment-argument-action function.*

$F_{E\Omega\Sigma}: E \times \Omega \times T \mapsto \Sigma \times T$

Let us now comment upon these sets and functions.

The function $F_{E\Sigma}$ allows us to define the argumentation relation which can exist between two commitment contents, i.e. a defense, an attack, a justification or a contradiction relation. For example:

$F_{E\Sigma}(SC(Ag_1, Ag_2, t_i, \varphi_i), SC(Ag_1, Ag_2, t_j, \varphi_j)) = (Defend\text{-}content, t_k)$

This means that the content of the commitment identified by t_i (called *source* of the defense relation) defends the content of the commitment identified by t_j (called *target* of the defense relation). The time unit t_k, associated with the defense relation, is the time at which this defense has occurred.

Schematically, the function $F_{E\Sigma}$ is presented in the following way (Figure 6.1):

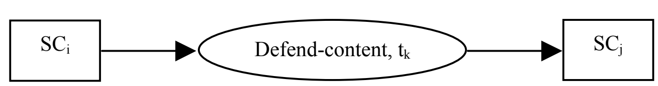


Figure 6.1. The function $F_{E\Sigma}$

In all the figures of this chapter, a social commitment identified by t_i will be denoted SC_i.

The function $F_{E\Sigma\Sigma}$ allows us to define an argumentation relation on another argumentation relation. For example:

$F_{E\Sigma\Sigma}(SC(Ag_1, Ag_2, t_i, \varphi_i), Defend\text{-}content, t_k) = (Attack\text{-}content, t_l)$

This relation points out that the content of the commitment identified by t_i attacks at time t_l the content of a defense relation that occurred at time t_k. This defense relation is defined using the function $F_{E\Sigma}$. The content of an argumentation relation is the content of the argument used in this relation.
Schematically, we present the function $F_{E\Sigma\Sigma}$ in the following way (Figure 6.2):

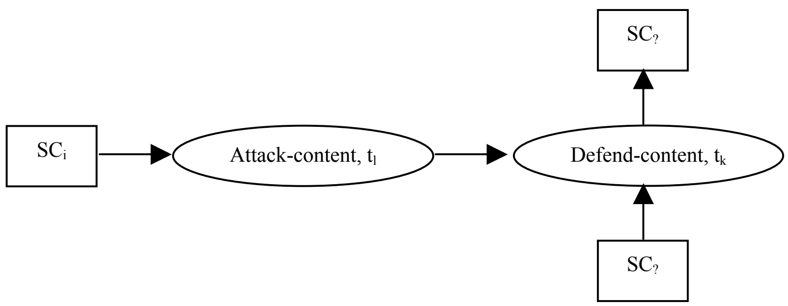


Figure 6.2. The function $F_{E\Sigma\Sigma}$

The function F_{Ω} allows us to define a set of creation and positioning actions (acceptance, refusal, etc.) performed by an agent on a commitment content. For example:

$F_{\Omega}(Ag_1, SC(Ag_2, Ag_1, t_i, \varphi_i)) = \{(Accept\text{-}content, t_k)\}$

This reflects the acceptance at moment t_k of the content related to the commitment identified by t_i.
Schematically, we present the function F_{Ω} as follows (Figure 6.3):

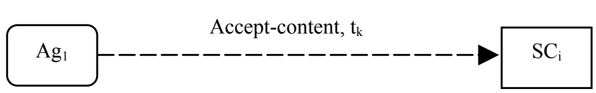


Figure 6.3. The function F_{Ω}

The function $F_{A\Sigma\Omega}$ allows an agent to take position by accepting or refusing an argumentation relation. For instance:

$F_{A\Sigma\Omega}(Ag_1, Defend\text{-}content, t_k) = \{(Refuse\text{-}content, t_l)\}$

This means that the agent Ag_1 refuses at time t_l the defense relation which is defined by the function $F_{E\Sigma}$. The defense relation has occurred at time t_k.
The function $F_{A\Sigma\Omega}$ is presented as follows (Figure 6.4):

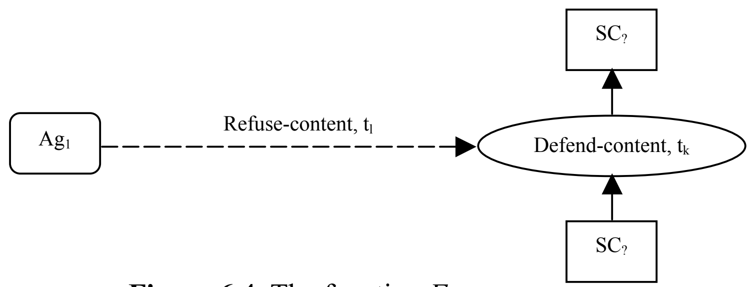


Figure 6.4. The function $F_{A\Sigma\Omega}$

The function $F_{A\Omega\Omega}$ allows an agent to position itself relative to a positioning action by accepting it, refusing it, challenging it, withdrawing it or reactivating it. The positioning action on which an agent can take positions can be defined by the function F_{Ω} or the function $F_{A\Sigma\Omega}$. For instance:

$F_{A\Omega\Omega}$ $(Ag_1, \textit{Refuse-content}, t_k) = \{(\textit{Challenge-content}, t_l)\}$

This example shows the case in which the agent Ag_1 challenges at time t_l a refusal action that occurred at time t_k. This refusal action is defined by the function F_{Ω}.
Schematically, the function $F_{A\Omega\Omega}$ is illustrated as follows (Figure 6.5):

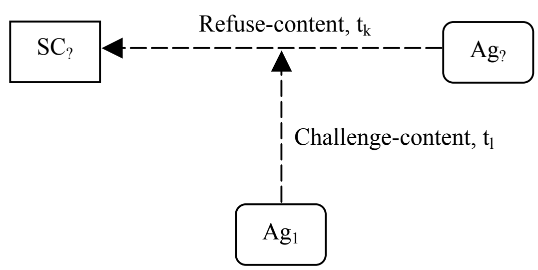


Figure 6.5. The function $F_{A\Omega\Omega}$

The function $F_{E\Omega\Sigma}$ allows us to define an argumentation relation binding a commitment to a creation or a positioning action. This action is defined by the function F_{Ω}. For example:

$F_{E\Omega\Sigma}$ $(SC(Ag_1, Ag_2, t_i, \varphi_i), \textit{Refuse-content}, t_k) = (\textit{Defend-content}, t_l)$

This example highlights the case in which the content of the commitment identified by t_i defends at time t_l the refusal action that occurred at time t_k. The refusal action is defined by the function F_{Ω}.

The graphical representation of this function is shown as follows (Figure 6.6):

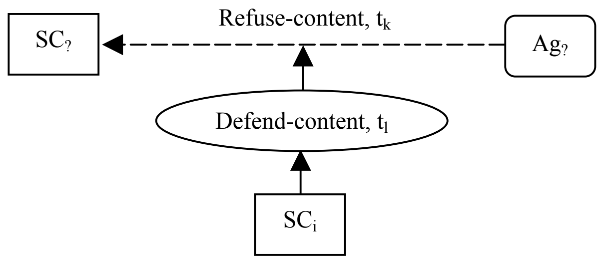


Figure 6.6. The function $F_{E\Omega\Sigma}$

6.3 Example

In this section, we show how to represent a dialogue between two agents using the CAN framework. We use the *conceptual graphs notation* (CG) proposed by Sowa (1984) in order to describe the propositional contents of commitments. Conceptual graphs are a system of logic and a knowledge representation language consisting of concepts and relations between these concepts. They are labeled graphs in which concept nodes are connected by relation nodes. With their direct mapping to natural language, CG serve as an intermediate language for translating computer-oriented formalisms to and from natural languages. A concept is represented by a type (ex. PERSON) and a referent (ex. john) and denoted [TYPE: Referent] (ex. [PERSON: John]). A conceptual relation links two concepts and is represented between brackets. When representing natural language sentences, case-relations are normally used. Examples are: AGNT (agent), PTNT (patient), OBJ (object), CHRC (characteristic), PTIM (point in time). The advantage of CG over predicate calculus is that they can be used to represent the literal meaning of utterances, without ambiguities, and in a logically precise form.

Before considering the example, we introduce the following notation: $S_{(H,h)}$ denoting the set of different states of an argument (*H*, *h*). $S_{(H,h)}$ is a finite and ordered set. The ordering relation $\prec$ between the elements of this set is defined as follows:

Definition 6.2 $\forall s_1, s_2 \in S_{(H,h)},\ \ s_1 \prec s_2$ iff the argument (*H*, *h*) was in state s_1 before to be in state s_2.

The current state of an argument (*H*, *h*) is the biggest element of the set $S_{(H,h)}$ according to the ordering relation $\prec$.

Let us consider the following dialogue *D1*:

$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0)$: The disease M is not genetic.
$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1)$: Why?
$SA(I_2, Ag_1, Ag_2, t_{u2}, U_2)$: Because it does not appear at birth.
$SA(I_3, Ag_2, Ag_1, t_{u3}, U_3)$: A disease which does not appear at birth can be genetic as well.
$SA(I_4, Ag_1, Ag_2, t_{u4}, U_4)$: How?
$SA(I_5, Ag_2, Ag_1, t_{u5}, U_5)$: It can be due to a genetic anomaly in the DNA appearing at a certain age.
$SA(I_6, Ag_1, Ag_2, t_{u6}, U_6)$: It is true, you are right.

With its speech act identified by I_0, agent Ag_1 creates, as explained in Chapter 5, a propositional commitment, i.e.:

$$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0) =_{def}$$
$$Create(Ag_1, t_{u0}, PC(Ag_1, Ag_2, t_0, p_0))$$
$$S^{t_0} = \{\textbf{\textit{active}}\}$$
$$S^{t_0}_{content} = \{\textbf{\textit{submitted}}\}$$

where $PC(Ag_1, Ag_2, t_0, p_0)$ is the initial commitment of the dialogue and p_0 is the propositional content which can be described by the following CG:

¬[[DISEASE : M]→(CHRC)→[GENETIC]]

In the CAN formalism, this speech act results in the function:

$$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, t_0, p_0)) = \{(Create, t_{u0})\}$$

Thereafter, agent Ag_2 performs the speech act identified by I_1 and takes position on the content of $PC(Ag_1, Ag_2, t_0, p_0)$ by challenging it. Thus, "challenged" becomes the current state of the commitment. Hence, we have:

$$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1) =_{def}$$
$$Challenge\text{-}content(Ag_2, t_{u0}, PC(Ag_1, Ag_2, p_0))$$
$$S^{t_0} = \{\textbf{\textit{active}}\}$$
$$S^{t_0}_{content} = \{submitted, \textbf{\textit{challenged}}\}$$

In the CAN formalism, this speech act results in the function:

$$F_{\Omega}(Ag_2, PC(Ag_1, Ag_2, t_0, p_0)) = \{(Challenge\text{-}content, t_{u1})\}$$

Then, agent Ag_1 justifies the propositional content p_0 of its commitment by performing the speech act identified by I_2. Hence, it creates another commitment $PC(Ag_1, Ag_2, t_1, p_1)$. Thus, "justified" becomes the current state of the commitment identified by t_0. We have:

$SA(I_2, Ag_1, Ag_2, t_{u2}, U_2) =_{def}$
$Justify\text{-}content(Ag_1, t_{u2}, PC(Ag_1, Ag_2, t_0, p_0))$
$\& \; Create(Ag_1, t_{u2}, PC(Ag_1, Ag_2, t_1, p_1))$

$$S^{t_0} = \{\textbf{active}\}$$

$$S^{t_0}_{content} = \{submitted, challenged, \textbf{justified}\}$$

$$S^{t_1} = \{\textbf{active}\}$$

$$S^{t_1}_{content} = \{\textbf{submitted}\}$$

where p_1 is described by the following CG:

¬[[DISEASE : M]←(AGNT)←[APPEAR]→(PTIM)→[BIRTH]]

The Ag_1's knowledge base contains the arguments (p_1, p_0) and (p_1, p_1). Thus, in argumentation terms, agent Ag_1 presents its argument (p_1, p_0). We have:

$Arg(Ag_1, p_1, Justify\text{-}content(Ag_1, t_{u0}, PC(Ag_1, Ag_2, t_0, p_0))$

This is represented in the CAN formalism by the functions:

$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, t_1, p_1)) = \{(Create, t_{u2})\}$,
$F_{E\Sigma}(\, PC(Ag_1, Ag_2, t_1, p_1), PC(Ag_1, Ag_2, t_0, p_0)) = (Justify\text{-}content, t_{u2})$

By the speech act identified by I_3, agent Ag_2 refuses Ag_1's argument. Then, it creates a new commitment $PC(Ag_2, Ag_1, t_2, p_2)$. We have:

$SA(I_3, Ag_2, Ag_1, t_{u3}, U_3\,) =_{def}$
$Refuse\text{-}content(Ag_2, t_{u3,}\, Arg(Ag_1, p_1, Justify\text{-}content(Ag_1, t_{u0}, PC(Ag_1, Ag_2, t_0, p_0))))$
$\& \; Create(Ag_2, t_{u3}, PC(Ag_2, Ag_1, t_2, p_2))$

$$S_{(p_1, p_0)} = \{\textbf{refused}\}$$

$$S^{t_2} = \{\textbf{active}\}$$

$$S^{t_2}_{content} = \{\textbf{submitted}\}$$

where the content p_2 is described by the following CG[4]:

¬[¬[[DISEASE :*x]←(AGNT)←[APPEAR]→(PTIM)→BIRTH]]
∧ [[*x]→(CHRC)→[GENETIC]]]

This is represented in the CAN formalism by the functions:

---

[4] To get this graph, we use the rule:
$p \Rightarrow q \equiv \neg(p \wedge \neg q)$, with $p = \neg$("there is a disease that appears at birth") and $q = \neg$("this disease is genetic").
Note that in the formula, *x is a mark of coreference which appears in the referent part of a concept.

$F_{A\Sigma\Omega}(Ag_2, Justify\text{-}content, t_{u2}) = \{(Refuse\text{-}content, t_{u3})\}$,
$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, p_2)) = \{(Create, t_{u3})\}$

Agent Ag_1's speech act, identified by I_4, challenges the content of the commitment identified by t_2. This allows us to change the content for the "challenged" state:

$SA(I_4, Ag_1, Ag_2, t_{u4}, U_4) =_{def}$
$Challenge\text{-}content(Ag_1, t_{u4}, PC(Ag_2, Ag_1, t_2, p_2))$
$S^{t2} = \{\textbf{active}\}$
$S^{t2}_{content} = \{submitted, \textbf{challenged}\}$

In the CAN formalism, this results in the function:

$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_2, p_2)) = \{(Challenge\text{-}content, t_{u4})\}$

Then, agent Ag_2 justifies the content of its commitment $PC(Ag_2, Ag_1, t_2, p_2)$ by performing the speech act identified by I_5. It then creates another commitment $PC(Ag_1, Ag_2, t_3, p_3)$. Thus, "Justified" becomes the current state of $PC(Ag_2, Ag_1, t_2, p_2)$. We have:

$SA(I_5, Ag_2, Ag_1, t_{u5}, U_5) =_{def}$
$Justify\text{-}content(Ag_2, t_{u5}, PC(Ag_2, Ag_1, t_2, p_2))$
$\& \; Create(Ag_2, t_{u5}, PC(Ag_2, Ag_1, t_3, p_3))$
$S^{t2} = \{\textbf{active}\}$
$S^{t2}_{content} = \{(submitted, challenged, \textbf{justified}\}$
$S^{t3} = \{\textbf{active}\}$
$S^{t3}_{content} = \{\textbf{submitted}\}$

where the content p_3 is described by the following CG:

[[ANOMALY-DNA : *x]-
(AGNT)←[CAUSE]→(PTNT)→[DISEASE : M]]
[[*x]←(AGNT)←[APPEAR]→(PTIM)→[AGE : @certain]]
In argumentation terms, agent Ag_2 presents its argument (p_3, p_2). Thus, we have:

$Arg(Ag_2, p_3, Justify\text{-}content(Ag_2, t_{u5}, PC(Ag_2, Ag_1, t_2, p_2))$

In the CAN formalism, this results in the following functions:

$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, p_3)) = \{(Create, t_{u5})\}$,
$F_{E\Sigma}(PC(Ag_2, Ag_1, t_3, p_3), PC(Ag_2, Ag_1, t_2, p_2)) = (Justify\text{-}content, t_{u5})$

Agent Ag_2's speech act, identified by I_6, reflects Ag_2's acceptance of both the content of the commitment identified by t_3 and the argument defending it. Thus, "Accepted" is the final state of this commitment. We have:

$SA(I_6, Ag_1, Ag_2, t_{u6}, U_6) =_{def}$
$\quad Accept\text{-}content(Ag_1, t_{u6}, Arg(Ag_2, p_3, Justify\text{-}content(Ag_2, t_{u5}, PC(Ag_2, Ag_1, t_2, p_2))))$
$\quad \& \; Accept\text{-}content(Ag_1, t_{u6}, PC(Ag_2, Ag_1, t_3, p_3))$

$$S_{(p_3, p_2)} = \{\boldsymbol{accepted}\}$$

$$S^{t_3} = \{active, \boldsymbol{satisfied}\}$$

$$S^{t_3}_{content} = \{submitted, \boldsymbol{accepted}\}$$

In the CAN formalism, this is represented by the functions:

$F_{A\Sigma\Omega}(Ag_1, Justify\text{-}content, t_{u5}) = \{(Accept\text{-}content, t_{u6})\}$,
$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_3, p_3)) = \{(Accept\text{-}content, t_{u6})\}$

To summarize, the dialogue *D1* can be represented by the following CAN:
$<A, E, PC(Ag_1, Ag_2, t_0, p_0), T, \Omega, \Sigma, F_{E\Sigma}, F_{E\Sigma\Sigma}, F_{\Omega}, F_{A\Sigma\Omega}, F_{A\Omega\Omega}, F_{E\Omega\Sigma}>$ such that:

$A = \{Ag_1, Ag_2\}$
$E = \{PC(Ag_1, Ag_2, t_0, p_0), PC(Ag_1, Ag_2, t_1, p_1), PC(Ag_2, Ag_1, t_2, p_2), PC(Ag_2, Ag_1, t_3, p_3)\}$
$T = \{t_{u0}, \ldots, t_{u6}\}$
$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, t_0, p_0)) = \{(Create, t_{u0})\}$
$F_{\Omega}(Ag_2, PC(Ag_1, Ag_2, t_0, p_0)) = \{(Challenge\text{-}content, t_{u1})\}$
$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, t_1, p_1)) = \{(Create, t_{u2})\}$
$F_{E\Sigma}(PC(Ag_1, Ag_2, t_1, p_1), PC(Ag_1, Ag_2, t_0, p_0)) = (Justify\text{-}content, t_{u2})$
$F_{A\Sigma\Omega}(Ag_2, Justify\text{-}content, t_{u2}) = \{(Refuse\text{-}content, t_{u3})\}$
$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, t_2, p_2)) = \{(Create, t_{u3})\}$
$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_2, p_2)) = \{(Challenge\text{-}content, t_{u4})\}$
$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, t_3, p_3)) = \{(Create, t_{u5})\}$
$F_{E\Sigma}(PC(Ag_2, Ag_1, t_3, p_3), PC(Ag_2, Ag_1, t_2, p_2)) = (Justify\text{-}content, t_{u5})$
$F_{A\Sigma\Omega}(Ag_1, Justify\text{-}content, t_{u5}) = \{(Accept\text{-}content, t_{u6})\}$
$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_3, p_3)) = \{(Accept\text{-}content, t_{u6})\}$

Figure 6.7 shows the graphical representation of the network.

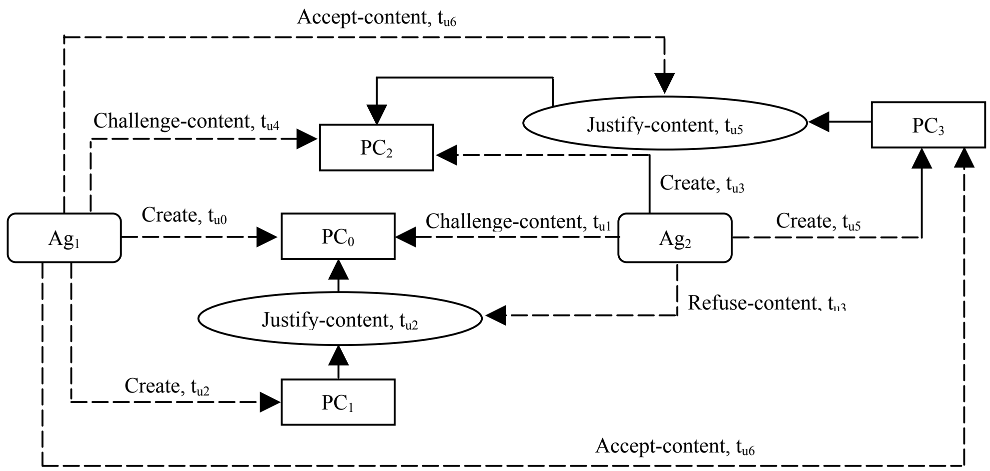


Figure 6.7. The network representing the dialogue *D1*

6.4 CAN: a Means of Inter-Agent Communication

So far, we have shown how the CAN formalism enables us to illustrate the connectedness of speech acts performed by agents in a conversation. In the previous section's example, we started from an existing dialogue, which we examined and modeled it using a CAN. This highlights a process that enables us to analyze a conversation using the CAN formalism. However, our formalism also provides a means for agents to take part in conversations.

Agents can jointly build the network that represents their conversation as it progresses. This allows agents:

1- To make sure at any time that the conversation is consistent;

2- To determine which speech act to perform on the basis of the current state of the conversation, using an argumentation system and other cognitive elements.

Consistency is ensured by the relationships existing between various commitments, diverse argumentation relations and different actions (creation, acceptance, fulfillment, etc.). A speech act is consistent with the rest of the conversation if it leads to the creation of a new commitment related to another commitment through an argumentation relation, or if it makes it possible to take position on a commitment, on an argumentation relation or on an action (i.e. creation, refusal, etc.). Moreover, the agent must know everything about the current state of the conversation in order to determine its next speech act. For example, when an agent creates a commitment and/or an argumentation relation, the other agent may decide to act on what has been created by accepting it, by refusing it, or by challenging it, depending on its argumentation system. Similarly, when an agent finds that its commitment, argument or action is being challenged, it must create a commitment in order to justify it. The network is built as the conversation progresses. This process differs from the one used to analyze a conversation. Therefore, agents use a dynamic process in order to build the network while taking part in the conversation.

In order to illustrate this way of using the CAN formalism, we revisit the example of Section 6.3 and demonstrate how agents build the network piece by piece while performing their speech acts. By doing that, agents are able to continue the conversation. The rules for building a CAN are the constraints specified in the axioms presented in Chapter 5. These axioms specify how agents can perform communicative acts according to there argumentation systems. The Ag_1's knowledge base contains the arguments (p_1, p_0), (p_1, p_1), and (p_3, p_3). The Ag_2's knowledge base contains the arguments (p_3, p_2) and (p_3, p_3).

Let us simulate the conversation of agents Ag_1 and Ag_2 using the CAN approach. Agent Ag_1 decides to start the conversation about a particular topic p_0 that interests it (the underlying mechanism related to this choice belongs to the cognitive layer that is not considered here (see our agent architecture in Section 5.6 of Chapter 5)). Hence, Ag_1 creates a propositional commitment whose content is p_0 since it has an argument supporting it, i.e.:

$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, t_0, p_0)) = \{(Create, t_{u0})\}$

This corresponds to the speech act identified by I_0:
$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0)$: The disease M is not genetic.

Then, agent Ag_2 decides to take position on the content of $PC(Ag_1, Ag_2, t_0, p_0)$ by challenging it since it does not have any argument in favor or against it. As a matter of fact, Ag_2 wants to know which Ag_1's argument supports the content of this commitment. Therefore, Ag_2 performs the action corresponding to the speech act identified by I_1:

$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1)$: Why?

$F_{\Omega}(Ag_2, PC(Ag_1, Ag_2, p_0)) = \{(Challenge\text{-}content, t_{u1})\}$

We notice here that as for commitment attempts (Chapter 5, Axiom *A*3), we cannot verify wether Ag_2 has an argument for or against p_0 or not because this aspect is related to its private internal state.

Now, Ag_1 must defend its proposition: it creates the commitment $PC(Ag_1, Ag_2, t_1, p_1)$ whose content justifies the content of $PC(Ag_1, Ag_2, t_0, p_0)$. In doing so, this agent performs the action corresponding to the speech act identified by I_2:

$SA(I_2, Ag_1, Ag_2, t_{u2}, U_2)$: Because it does not appear at birth.

$F_{\Omega}(Ag_1, PC(Ag_1, Ag_2, p_1)) = \{(Create, t_{u2})\}$
$F_{E\Sigma}(PC(Ag_1, Ag_2, t_1, p_1), PC(Ag_1, Ag_2, t_0, p_0)) = (Justify\text{-}content, t_{u2})$

Ag_2 has an argument against the justification relation. Consequently, it refuses it by creating the commitment $PC(Ag_2, Ag_1, t_2, p_2)$. It performs the action corresponding to the speech act identified by I_3:

$SA(I_3, Ag_2, Ag_1, t_{u3}, U_3)$: A disease which does not appear at birth can be genetic as well.

$F_{A\Sigma\Omega}(Ag_2, Justify\text{-}content, t_{u2}) = \{(Refuse\text{-}content, t_{u3})\}$
$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, t_2, p_2)) = \{(Create, t_{u3})\}$

Because agent Ag_1 does not have any argument for or against p_2, it challenges the content of $PC(Ag_2, Ag_1, t_2, p_2)$ using its argumentation system. By doing that, it performs the action corresponding to the speech act identified by I_4:

$SA(I_4, Ag_1, Ag_2, t_{u4}, U_4)$: How?

$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_2, p_2)) = \{(Challenge\text{-}content, t_{u4})\}$

The content of Ag_2's commitment $PC(Ag_2, Ag_1, t_2, p_2)$ being challenged. Therefore, agent Ag_2 must try to justify it. Because its knowledge base contains the argument (p_3, p_2), it creates the commitment $PC(Ag_2, Ag_1, t_3, p_3)$ and performs the actions corresponding to the speech act identified by I_5:

$SA(I_5, Ag_2, Ag_1, t_{u5}, U_5)$: It can be due to a genetic anomaly in the DNA appearing at a certain age.

$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, t_3, p_3)) = \{(Create, t_{u5})\}$
$F_{E\Sigma}(PC(Ag_2, Ag_1, t_3, p_3), PC(Ag_2, Ag_1, t_2, p_2)) = (Justify\text{-}content, t_{u5})$

Thereafter, because the Ag_1's knowledge base contains an argument for p_3, it accepts the content of $PC(Ag_2, Ag_1, t_3, p_3)$ and the argumentation relation $(Justify\text{-}content, t_{u5})$ using its argumentation system. It performs the actions corresponding to the speech act identified by I_6:

$SA(I_6, Ag_1, Ag_2, t_{u6}, U_6)$: It is true, you are right.

$F_{A\Sigma\Omega}(Ag_1, Justify\text{-}content, t_{u5}) = \{(Accept\text{-}content, t_{u6})\}$
$F_{\Omega}(Ag_1, PC(Ag_2, Ag_1, t_3, p_3)) = \{(Accept\text{-}content, t_{u6})\}$

6.5 Other Examples

In the following examples, we give the final version of the networks without illustrating the steps that led to their construction. Moreover, for simplicity, we do not describe the content of commitments.

The example presented in Section 6.3 illustrated the case in which an agent takes position on a commitment and on an argumentation relation. The following example of dialogue (*D2*) illustrates the case in which an agent takes position on a creation action.

$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0)$: I will travel to the Himalayas.
$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1)$: Why do you tell me that?
$SA(I_2, Ag_1, Ag_2, t_{u2}, U_2)$: It is only to inform you.
$SA(I_3, Ag_2, Ag_1, t_{u3}, U_3)$: Ok, thank you.

The network associated with this dialogue is:
$<A, E, AC(Ag_1, Ag_2, t_0, (\alpha, p_0)), T, \Omega, \Sigma, F_{E\Sigma}, F_{E\Sigma\Sigma}, F_\Omega, F_{A\Sigma\Omega}, F_{A\Omega\Omega}, F_{E\Omega\Sigma}>$ such that:
$A = \{Ag_1, Ag_2\}$
$E = \{ AC(Ag_1, Ag_2, t_0, (\alpha, p_0)), PC(Ag_1, Ag_2, t_1, p_1)\}$
$T = \{t_{u0}, \ldots, t_{u3}\}$

$F_\Omega(Ag_1, AC(Ag_1, Ag_2, t_0, (\alpha, p_0)) = \{(Create, t_{u0})\}$
$F_{A\Omega\Omega}(Ag_2, Create, t_{u0}) = \{(Challenge\text{-}content, t_{u1})\}$
$F_\Omega(Ag_1, PC(Ag_1, Ag_2, t_1, p_1)) = \{(Create, t_{u2})\}$
$F_{E\Omega\Sigma}(PC(Ag_1, Ag_2, t_1, p_1), Create, t_{u0}) = (Justify\text{-}content, t_{u2})$
$F_\Omega(Ag_2, PC(Ag_1, Ag_2, t_1, p_1)) = \{(Accept\text{-}content, t_{u3})\}$

The graphical representation of this network is illustrated by Figure 6.8.

Agent Ag_1 creates an action commitment $AC(Ag_1, Ag_2, t_0, (\alpha, p_0))$ (it is committed to traveling to the Himalayas) by performing the speech act identified by I_0. Thereafter, agent Ag_2 challenges the creation action of this commitment by performing the speech act identified by I_1. In order to justify its creation action of $AC(Ag_1, Ag_2, t_0, (\alpha, p_0))$, Ag_1 creates a propositional commitment $PC(Ag_1, Ag_2, t_1, p_1)$ by performing the speech act identified by I_2. Finally, Ag_2 accepts the content of this commitment by performing the speech act identified by I_3.

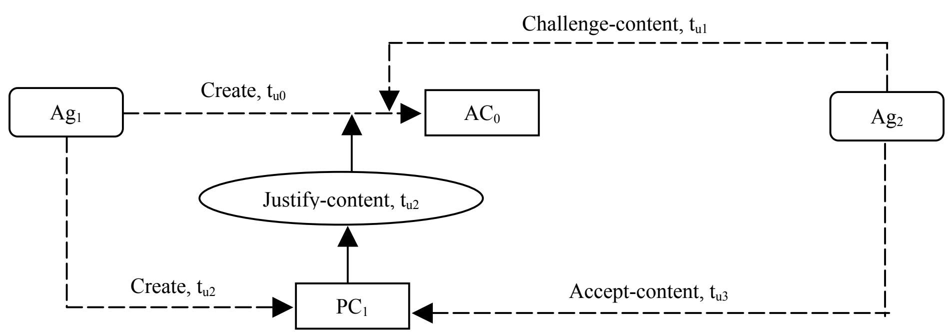


Figure 6.8. The network representing the dialogue *D2*

The CAN formalism also allows us to manage commitment attempts. Dialogues *D3* and *D4* illustrate respectively the acceptance and the refusal of a commitment attempt.

Dialogue *D3*:

$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0)$: Can you drive me to the airport at 5PM?
$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1)$: Yes, I can.
$SA(I_2, Ag_2, Ag_1, t_{u2}, U_2)$: I will be available at 5PM.

The network associated with this dialogue is:
$<A, E, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0)), T, \Omega, \Sigma, F_{E\Sigma}, F_{E\Sigma\Sigma}, F_{\Omega}, F_{A\Sigma\Omega}, F_{A\Omega\Omega}, F_{E\Omega\Sigma}>$ such that:
$A = \{Ag_1, Ag_2\}$
$E = \{ ACT(Ag_1, Ag_2, t_0, (\alpha, p_0)), AC(Ag_2, Ag_1, t_1, (\alpha, p_0)), PC(Ag_2, Ag_1, t_2, p_1)\}$
$T = \{t_{u0}, t_{u1}, t_{u2}\}$

$F_{\Omega}(Ag_1, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))) = \{(Create, t_{u0})\}$
$F_{\Omega}(Ag_2, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))) = \{(Accept\text{-}content, t_{u1})\}$
$F_{\Omega}(Ag_2, AC(Ag_2, Ag_1, t_1, (\alpha, p_0))) = \{(Create, t_{u1})\}$
$F_{\Omega}(Ag_2, PC(Ag_2, Ag_1, t_2, p_1)) = \{(Create, t_{u2})\}$
$F_{E\Sigma}(PC(Ag_2, Ag_1, t_2, p_1), AC(Ag_2, Ag_1, t_1, (\alpha, p_0))) = (Justify\text{-}content, t_{u2})$

The graphical representation of this network is illustrated by Figure 6.9.

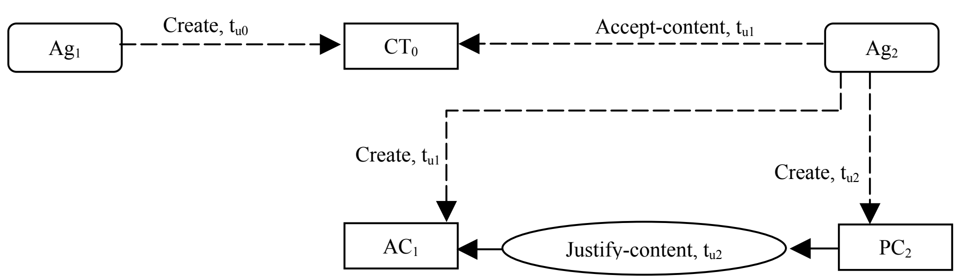


Figure 6.9. The network representing the dialogue *D3*

Agent Ag_1 creates a commitment attempt $ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))$ about an action α by performing the speech act identified by I_0. Agent Ag_2 accepts this commitment by performing the speech act identified by I_1. Therefore, it creates the action commitment $AC(Ag_2, Ag_1, t_1, (\alpha, p_0))$ (it commits to drive agent Ag_1 to the airport at 5PM). Thereafter, Ag_2 creates the propositional commitment $PC(Ag_2, Ag_1, t_2, p_1)$ that supports the content p_0 by performing the speech act identified by I_2.

Dialogue *D4*:

$SA(I_0, Ag_1, Ag_2, t_{u0}, U_0)$: Can you drive me to the airport at 5PM?
$SA(I_1, Ag_2, Ag_1, t_{u1}, U_1)$: No, I cannot.
$SA(I_2, Ag_1, Ag_2, t_{u2}, U_2)$: Why not?
$SA(I_3, Ag_2, Ag_1, t_{u3}, U_3)$: Because I have a meeting at 5PM.
$SA(I_4, Ag_1, Ag_2, t_{u4}, U_4)$: Ok, thank you.

The network associated with this dialogue is:
$<A, E, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0)), T, \Omega, \Sigma, F_{E\Sigma}, F_{E\Sigma\Sigma}, F_{\Omega}, F_{A\Sigma\Omega}, F_{A\Omega\Omega}, F_{E\Omega\Sigma}>$
such that:
$A = \{Ag_1, Ag_2\}$

$E = \{ACT(Ag_1, Ag_2, t_0, (\alpha, p_0)), PC(Ag_2, Ag_1, t_1, \neg p_0), PC(Ag_2, Ag_1, t_2, p_1)\}$
$T = \{t_{u0}, \ldots, t_{u3}\}$
$F_\Omega(Ag_1, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))) = \{(Create, t_{u0})\}$
$F_\Omega(Ag_2, ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))) = \{(Refuse\text{-}content, t_{u1})\}$
$F_\Omega(Ag_2, PC(Ag_2, Ag_1, t_1, \neg p_0)) = \{(Create, t_{u1})\}$
$F_\Omega(Ag_1, PC(Ag_2, Ag_1, t_1, \neg p_0)) = \{(Challenge\text{-}content, t_{u2})\}$
$F_\Omega(Ag_2, PC(Ag_2, Ag_1, t_2, p_1)) = \{(Create, t_{u3})\}$
$F_{E\Sigma}(PC(Ag_2, Ag_1, t_2, p_1), PC(Ag_2, Ag_1, t_1, \neg p_0)) = (Justify\text{-}content, t_{u3})$

The graphical representation of this network is illustrated by Figure 6.10.

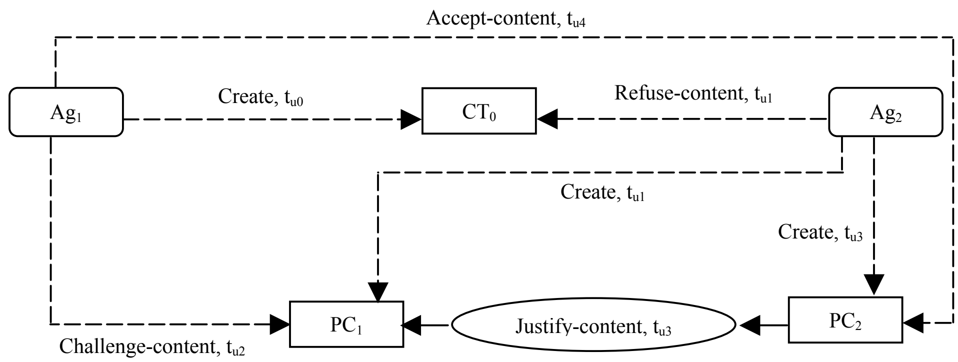


Figure 6.10. The network representing the dialogue *D4*

As a result of refusing the commitment attempt $ACT(Ag_1, Ag_2, t_0, (\alpha, p_0))$ by performing the speech act identified by I_1, agent Ag_2 creates the propositional commitment $PC(Ag_2, Ag_1, t_1, \neg p_0)$. By performing the speech act identified by I_2, agent Ag_1 challenges the content of this commitment. Therefore, Ag_2 creates the propositional commitment $PC(Ag_2, Ag_1, t_2, p_1)$, by performing the speech act identified by I_3, in order to justify $PC(Ag_2, Ag_1, t_1, \neg p_0)$.

6.6 CAN and Representation of Conversations

So far, we have shown how the CAN formalism allows us to represent conversations by illustrating the connectedness of speech acts performed by agents. However, we did not show if it can represent any coherent conversation. To do this we have to provide a mathematical demonstration. The purpose is to show that the formalism is sufficient to handle any argumentative conversation for communication between software agents. An argumentative conversation is a conversation that contains argumentation relations in order to achieve a goal (for example a persuasion or a negotiation goal). First, we have to define what is a conversation and what is a coherent conversation. For us, a conversation is a sequence of utterances (i.e. a sequence of speech acts). A coherent conversation is a conversation in which there is a positioning relation or an argumentation relation between the utterances. For example, if an agent Ag_1 performs a speech act whose content is p, and another agent Ag_2 performs another speech act in which it accepts, refuses, challenges, attacks, etc. p, then, this part of the conversation is considered as coherent. However, if Ag_2 performs a speech act whose content is q without any positioning or argumentation relation between p and q, then, the conversation is considered as incoherent.

In this section we show that the CAN formalism covers all the elements describing a conversation. We use for this purpose the following formal presentation due to (Günter, 1984).

Let A be a set of agents ($A = \{Ag_1, \ldots, Ag_n\}$), L be a set of well-formed expressions ($L = \{\varphi_0, \ldots, \varphi_m\}$), P be a set of designatory phrases ($P = \{p_0, \ldots, p_k\}$), and V be a set of performatives ($V = \{v_0, \ldots, v_l\}$). A conversation is a finite sequence of 4-tuples, each of which consists of: a name $Ag_i \in A$, a well-formed expression $\varphi_i \in L$, a performative verb $v_i \in V$, and a designatory phrase $p_i \in P$. The well-formed expressions represent the participants' statements. The term *sequence* highlights the temporal order in which these expressions are used. The names represent the participants in the conversation. The performative verb indicates the type of the speech act performed when using the expression. The designatory phrase identifies the speech act. Formally:

C is a conversation iff: there are a language L, a set A of participants, a set V of performative verbs, a set P of designatory phrases, and
$\exists n \in \mathbb{N}, \forall 1 \leq i \leq n, \exists Ag_i \in A, \exists \varphi_i \in L, \exists v_i \in V, \exists p_i \in P$ and
$C = ((Ag_1, \varphi_1, v_1, p_1), \ldots, (Ag_i, \varphi_i, v_i, p_i), \ldots, (Ag_n, \varphi_n, v_n, p_n))$.

The CAN formalism allows us to represent these various elements. The language L is used to describe the commitment content (for example predicate calculus or conceptual graphs). The expressions φ_i are thus represented by the commitment content φ. The set of the participants is the set A of the CAN formalism. The performative verbs and the designatory phrases are captured by the actions that agents perform on commitments and arguments. The sequence of the 4-tuples is modeled by the utterance times associated with the different actions in the CAN formalism. It is modeled by the set T of time units associated with the set of the actions Ω and to the set of the argumentation relations Σ (see Definition 6.1).

According to (Günter, 1984), a conversation can also highlight the goal of the accomplished actions. In the CAN formalism, this is illustrated by the fact that it is possible to justify not only a commitment content, but also a creation action of a commitment (see Definition 5.5 of Chapter 5).

Notation
We denote D the set of coherent conversations and R the set of commitment and argument networks. We denote a commitment and argument network which is associated with a coherent conversation C by $CAN(C)$ with C is an element of D and $CAN(C)$ an element of R.

Proposition 6.3 $\forall C \in D, \exists CAN(C) \in R$.

In other words, for any coherent conversation, there is always a CAN which represents it.

Proof

We use a proof by contradiction. A conversation C can be described in the simplest form as a sequence of utterances $U_0,\ldots,U_i,\ldots U_n$. Each utterance is associated with a participant Ag_j.

Let us assume that: $\exists C$ a coherent conversation such that $\nexists\ CAN(C)$. In other words, let us assume that there is a coherent conversation C such that no network can represent it. This implies the existence of an utterance U_i which one cannot represent in a network. Let $C' = U_0,\ldots,U_{i-1}$. Therefore the utterance U_i does not allow us to perform one of the following actions:

1- Creating a new commitment.
2- Taking position on a commitment of $CAN(C')$.
3- Taking position on an action of $CAN(C')$.
4- Taking position on an argumentation relation of $CAN(C')$.

It remains only two possibilities to interpret U_i:

1- Taking position on a commitment, an action or an argumentation relation which does not belong to $CAN(C')$. In this case the resulting conversation is not coherent because it highlights a positioning on an element which was not created. For example, challenging the content of a commitment which does not exist (see our definition of coherence above).

2- The utterance U_i cannot result in an element which can be supported by the elements of the CAN. This can be due to one of the two following reasons:
*Reason*1: The utterance U_i cannot lead to the creation of a commitment, a positioning action and / or an argumentation relation. This is false by definition.
*Reason*2: The positioning action reflected by U_i cannot be presented by one of the functions of the CAN (i.e. the functions: $F_{E\Sigma}$, $F_{E\Sigma\Sigma}$, F_{Ω}, $F_{A\Sigma\Omega}$, $F_{A\Omega\Omega}$, $F_{E\Omega\Sigma}$). This is false because it is possible to take position by *nesting*, *n* times, on a positioning action, or on an argumentation relation. The reason is that a positioning action of an unspecified order *X* is always represented by the Cartesian product: $\Omega \times T$.

Let us show this last issue by the illustration of Figure 6.11.

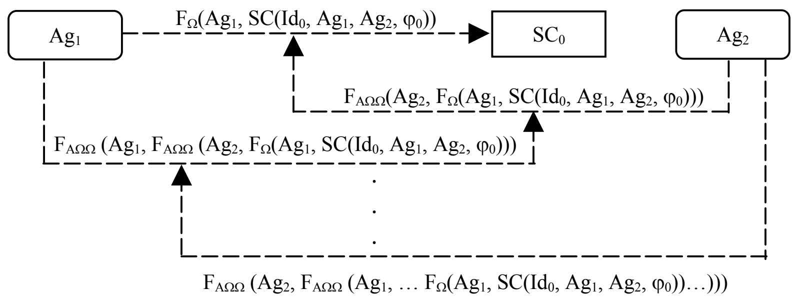


Figure 6.11. Illustration of nested positioning actions

Let Ω be the following set $\Omega = \{\Omega_0, \ldots, \Omega_m\}$. Using the definition of the function F_Ω we have:

$$F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_0, \varphi_0)) = (\Omega_0, t_1)$$

Using the definition of the function $F_{A\Omega\Omega}$ we obtain:

$$F_{A\Omega\Omega}(Ag_2, F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_0, \varphi_0))) = F_{A\Omega\Omega}(Ag_2, \Omega_0, t_1) = (\Omega_1, t_2)$$

Therefore, we obtain:

$$F_{A\Omega\Omega}(Ag_1, F_{A\Omega\Omega}(Ag_2, F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_0, \varphi_0)))) = F_{A\Omega\Omega}(Ag_1, \Omega_1, t_2) = (\Omega_2, t_3)$$

.
.
.

$$F_{A\Omega\Omega}(Ag_2, F_{A\Omega\Omega}(Ag_1, \ldots F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_0, \varphi_0))\underbrace{\ldots}_{n})) = F_{A\Omega\Omega}(Ag_2, \Omega_{n-2}, t_{n-1})$$
$$= (\Omega_{n-1}, t_n)$$

In the same way, one can show that it is always possible to define an argumentation relation on any argumentation relation created previously, considering that an argumentation relation of any order is represented by the Cartesian product: $\Sigma \times T$.

Therefore, the starting assumption is false. Thus, we proved that any coherent conversation can be represented by a CAN formalism.
□

Proposition 6.4 $\forall C \in D, \exists! CAN(C) \in N$.
In other words, for any coherent conversation, there is one and only one CAN which represents it.

Proof
The proof of this proposition is based on the proposition 6.1 and on the fact that any speech act can be interpreted in our approach in a unique way as an action performed on a commitment or on an argument. Because any action is presented by one and only one function, the CAN representing a conversation is unique.
□

In Section 6.2 we presented the structure of the CAN formalism, and we illustrated its construction process through the example of Section 6.3. In these two sections, we only highlighted the fact that the CAN formalism can be used to represent conversations. However, in the proposition 6.1, we showed generally that the CAN formalism is able to represent any coherent conversation, in particular by showing the falseness of the reason *Reason*2. The proposition is thus not a "petitio principii" since “nesting property” (see *Reason*2) is not an assumption in our proof. Our proof is rather a proof by construction because we showed that we can build a CAN for any coherent conversation.

This theoretical result is of great utility because it offers a formal framework to represent different types of conversations, for example, the conversation types proposed by Walton and Krabbe (1995).

6.7 Related Work

KQML was the first standard proposed to specify communications between agents (Finin et al., 1995). More recently, FIPA (1997, 1999, 2001a) proposed a new standard called FIPA-ACL. KQML and FIPA-ACL are both based on the mental approach. These two languages use protocols like those proposed by Pitt and Mamdani (2000) and the Contract Net (Smith, 1980) and the NetBill (Cox et al., 1995). These protocols define, in a fixed way, which sequences of moves are conventionally expected in a conversation. Protocols are often technically modeled as finite state machines that represent sequences of states and transitions and are usually too rigid to be used to model conversations between autonomous agents. In this context, the CAN formalism can allow the action sequences described by a protocol, but in a more flexible way. Contrary to protocols, agents using the CAN system do not follow a pre-planned sequence, but they reason in terms of commitments, arguments and relations between these two types of elements. In order to select the next communicative act to be performed, an agent reasons on the current state of the conversation using its argumentation system. This state is represented by the CAN framework and by the notion of commitment and argument state. In addition, protocols are semi-formally specified. However, the CAN framework is formally specified using a formal approach based on action and argumentation theories. These formal foundations allow us to prove some interesting properties like propositions 6.1 and 6.2. They also enable us to define a formal semantics and a verification method for agent communication using a model checking technique. Chapters 7 and 8 detail these two issues.

Several researchers proposed dialogue games in order to offer more flexibility (Dastani et al., 2000), (Maudet and Chaib-draa, 2002), (McBurney et al., 2002). The CAN formalism can be used to represent these dialogue games and to illustrate how various games can be combined in order to build complete conversation. In Chapter 9, we present a persuasion dialogue game protocol specified using our approach. Additionally, the CAN framework can be used not only as a specification tool but also as a means that agents can use in order to be able to effectively participate in coherent conversations.

Singh and Colombetti propose a commitment-based approach that emphasizes the importance of the social aspect of communication (Colombetti, 2000), (Singh, 1998, 2000). Singh's and Colombetti's work were focused on the definition of a semantics for speech acts. When considering the conversational aspect, Singh simply proposed the enhancement of the classical protocols (like those used in FIPA) by using commitments in order to ensure the compliance of the agents' behavior with the protocol. A participating agent can maintain a record of the commitments being created and modified. From these, the agent can determine the compliance of the other agents according to the given protocol. However, this approach is still not flexible and it does not indicate how agents can select the communicative acts. Colombetti proposed general conversational principles from which the structure of well-formed conversations should be derived. However, the way of implementing these principles is not specified. The management of commitments is only partially addressed in this approach.

On the basis of Singh's and Colombetti's proposals, Yolum and Singh (2002) developed a technique for specifying protocols in which actions' content is captured through agents' commitments. They provide operations and reasoning rules to capture the evolution of commitments. Using these rules, agents can reason about their actions. Chopra and Singh (2004) proposed a commitment-based formalism called *non-monotonic commitment machines* for representing multi-agent interaction protocols. This formalism specifies rules using *nonmonotonic causal logic*. These rules model the changes in the state of a protocol as a result of the performance of actions. The nonmonotonic causal logic in this formalism is used only to reason about actions in terms of whether an action can be the cause of another action. However, how agents can select actions using this reasoning mechanism is not addressed. In addition, the relation between this reasoning and private mental states of agents is not specified. In a similar way, Fornara and Colombetti (2003) proposed a method to define interaction protocols. This method is based on the specification of an interaction diagram (ID) specifying which actions can be performed under given conditions. The advantage of these approaches is that they are verifiable because they are based on public notions. They also allow us to represent the interaction dynamics through the allowed operations. Like these proposals, our approach and our CAN formalism are also based on commitments. However, our approach uses an argumentation theory which is more general that the nonmonotonic causal logic used in (Chopra and Singh, 2004). This is due to the fact that in our approach, agents can reason about commitments, commitment contents, and positioning actions in order to decide about the next act to be performed. This argumentation-based reasoning uses both the agents' mental states and the current state of the conversation. Our approach explicitly specifies how agents handle their commitments and how they take positions on other agents' commitments by using arguments. In addition, the operations we use in our pragmatic approach are different from the operations used in (Fornara and Colombetti, 2003), (Chopra and Singh, 2004), (Yolum and Singh, 2002). Finally, unlike the other formalisms, the CAN formalism can be used both to assist agents to communicate in a coherent way by representing the evolution of the conversation and to specify flexible protocols using, for example, the dialogue game approach.

Amgoud and her colleagues (2000a, 2000b, 2001) proposed to model dialogues using an argumentative approach and formal dialectics. Using MacKenzie's dialectical system (1979), they defined a certain number of dialogue rules and update rules for the different types of locutions supported by their dialogue model. These locutions are: assert, accept, question, challenge, request, promise and refuse. Dialogue rules define the protocol, while update rules capture the effect of the speech acts on the state of the dialogue. To reflect the dialogue dynamics, they use the concept of a *commitment store*. Each agent has its own commitment store accessible by all the other agents. These commitment stores contain only the moves which were performed. Therefore, they reflect only the dialogue history. In the same way, Parsons et al. (2003), McBurney (2002) and Sadri et al. (2001) proposed protocols based on an argumentative approach. These protocols are based on Walton and Krabbe's classification of dialogues and on formal dialectics. In these protocols, agents can argue about the truth of propositions. Agents can communicate both propositional statements and arguments about these statements. These protocols have the advantage of taking into account the capacity of agents to reason as well as their attitudes (confident, careful,…). Semantically, these protocols are specified by defining pre- and post-conditions for each locution. The main difference between these proposals and our work is that our approach formalizes a social aspect of agent interaction (represented by the notion of social commitments) and its relation to the agent reasoning using an argumentation theory. Thus, our approach is an hybrid one that is based on commitments and arguments. Another important difference is that argumentation-based protocols (McBurney, 2002), (McBurney et al., 2002), (Parsons et al., 2003) use moves from formal dialectics, whereas our approach uses an action theory to specify agents' speech acts as actions that these agents apply to commitments and to arguments. The semantics of these actions is defined in Chapter 7 using dynamic logic. By using these actions we can capture not only the locutions used in these protocols but also the argumentation actions represented in our framework by attack, defense, justify and contradict actions. In addition, in our approach, dynamics is reflected not only by the connectedness of the commitments resulting from the performed speech acts, but also by the concepts of the commitment state, the commitment content state and the argument state. The CAN formalism more clearly illustrates this dynamics in terms of actions on commitments and arguments. Moreover, unlike the CAN formalism, the notion of commitment store does not make it possible to distinguish the argumentation phases from the other phases and does not allow us to illustrate the positioning of an agent on an another agent's action.

Reed (1998) introduced the notion of dialogue frame as a model of inter-agent communication. He used this notion to present the dialogue types defined by Walton and Krabbe (1995): persuasion, negotiation, investigation, deliberation and information seeking. These types are represented by a set D as follows:

$$D = \{ < persuade, B >, < negotiate, C >, < inquire, B >, < deliberate, P >, < infoseek, B > \}$$

where B is a set of agent's beliefs, C a set of agent's contracts, and P a set of agent's plans.

Formally, a dialogue frame is a 4-tuple:

$$F = << t, \Delta > \in D, \tau \in \Delta, \left\{ u^{0}_{x_0 \rightarrow y_0, \ldots,} u^{n}_{x_n \rightarrow y_n} \right\} >$$

where t is the type of the dialogue frame, Δ is the set of beliefs, contrasts or plans, τ is the topic of the dialogue frame, x_0 and y_0 are the interlocutors and $u^{j}_{xj \rightarrow yj}$ refers to the jth utterance occurring in a dialogue between agents x_j and y_j such that ($x_j = y_{j+1}$ and $y_j = x_{j+1}$). A dialogue frame is of a particular type ($<t, \Delta> \in D$), and focused on a particular topic ($\tau \in \Delta$). For instance, a persuasion dialogue will be focused on a particular belief, a deliberation on a plan, and so on. Reed's approach makes it possible to illustrate the conversation dynamics only in terms of sequences of utterances. As an external representation, the CAN formalism is more complete than the concept of dialogue frame. In the CAN formalism, the dynamics is reflected by the actions that agents perform on commitments and arguments and by the argumentation relations existing between these commitments and arguments. The sequence of utterances is captured in our framework by the set T of time units that we associate with the various actions. In addition to being a means to analyze conversations, the CAN formalism provides agents with a means that enables them to participate in coherent conversations and to select their future moves. Like the dialogue frames, our formalism can represent any dialogue type. In Chapter 9, we present the example of the persuasion dialogue.

Chapter 7[*]

A Logical Model for Commitments and Arguments

In this chapter, we develop a semantics of the pragmatic approach proposed in Chapters 5 and 6. We propose a logical model based on CTL (Extended Computation Tree Logic) and on Dynamic logic that we call* $DCTL^*_{CAN}$. *This logical model addresses three basic elements: social commitments, actions that agents apply to these commitments and arguments that agents use to support their actions. The advantage of this logical model is to gather all these elements and the existing relations between them within the same framework. The semantics we develop here makes it possible to reflect the dynamics of agent communication. It also allows us to establish the important link between commitments as a deontic concept and arguments. On the one hand CTL* enables us to express all the temporal aspects related to the handling of commitments and arguments. On the other hand, dynamic logic enables us to capture the actions which agents are committed to perform.*

7.1 Introduction

In the domain of agent communication, semantics is one of the most important aspects particularly in the current context of open and interoperable multi-agent systems (MAS) (Chaib-draa and Dignum, 2002), (Dignum and Greaves, 2000). Although much significant research work was done in this field, for example (Singh, 2000), (Wooldridge, 2000), (Guerin and Pitt, 2001), (Amgoud et al., 2002), (Vericchio and Colombetti, 2003), the definition of a clear and global semantics (i.e. dealing with the various aspects of agent communication) is an objective yet to be reached.

While pragmatics deals with the way of using communication acts, semantics is interested in the meaning of these acts. Pragmatics is related to the dynamics of agent interactions and to the way of connecting the isolated acts to build complete conversations. Pragmatics was also addressed by many researchers, for example (Dastani et al., 2000), (Pitt and Mamdani,

[*] We would like to thank John-Jules Ch. Meyer, Frank Dignum, Mehdi Dastani, Henry Prakken and Broersen Jan from Utrecht University, Intelligent Systems Group, Josée Desharnais from Laval University, Department of Computer Science and Software Engineering, and Yves Lespérance from York University for the helpful discussions and for their useful comments about the logical model presented in this chapter. An extended abstract of this chapter is presented in AAMAS'04 (Bentahar et al., 2004e). Another version is presented in (Bentahar et al., 2004f).

2000), (Pasquier and Chaib-draa, 2003). However, little previous work tried to address these two facets of agent communication in the same framework, considering the difficulty of such a task. Even in this work, semantics and pragmatics are dealt with as a unique object of research whereas they are different in nature. In this context, we believe that the success of applications based on agent communication requires to address these two elements together but keeping them distinct.

The objective of this chapter is to develop the semantic part of our unified framework based on commitments and arguments for agent communication. Thus, the chapter deals with semantic issues in the approach proposed in Chapters 5 and 6 and the link with pragmatic ones. The semantics we define here addresses all the aspects that we use in our commitment and argument approach. This chapter presents two results: 1. it semantically establishes the link between commitments and arguments; 2. it uses a combination of temporal logic (CTL* with some additions) and a dynamic logic to define a complete and unambiguous semantics.

The rest of this chapter is organized as follows. In Section 7.2, we recall the taxonomy of social commitments we used in our pragmatic approach. In Sections 7.3 and 7.4, we present the syntax and the semantics of our logical model. In Section 7.5, we define some postulates. A discussion is presented in Section 7.6 and finally we conclude the chapter.

7.2 The Taxonomy of Social Commitments

In the following section, we briefly recall the taxonomy we presented in Chapter 5. We use this taxonomy in the logical model presented in this chapter.

***A. Absolute Commitments* (*ABC*)**

Absolute commitments are commitments whose fulfillment does not depend on any particular condition. Two types can be distinguished: propositional commitments and action commitments.

***A1. Propositional Commitments* (*PC*)**

Propositional commitments are related to the state of the world. They are generally, but not necessarily[5], expressed by assertives. They can be directed towards the past, the present, or the future.

***A2. Action Commitments* (*AC*)**

Action commitments (also called commitments to a course of action) are directed towards the present or the future and are related to actions that the debtor is committed to carrying out. The fulfillment and the lack of fulfillment of such commitments depend on the performance of the underlying action and the specified delay. This type of commitment is typically conveyed by promises.

---

[5] Propositional commitments can also be expressed by speech acts of declaratory and expressive types.

***B. Conditional Commitments* (*CC*)**

Absolute commitments do not consider conditions that may make relative the need for their fulfillment. However, in several cases, agents need to make commitments not in absolute terms but under given conditions. Another commitment type is therefore required. These commitments are said to be conditional. We distinguish between conditional commitments about propositions (***PCC***) and conditional commitments about actions (***ACC***). A conditional commitment about a proposition *p'* expresses the fact that if a condition *p* is true, then the creditor will be committed towards the debtor that *p'* is true.

***C. Commitment Attempts* (*CT*)**

The commitments described so far directly concern the debtor who commits either that a certain fact is true or that a certain action will be carried out. For example, these commitments do not allow us to explain the fact that an agent asks another one to be committed to carrying out an action (by a speech act of a directive type). To solve this problem, we propose the concept of commitment attempt. We consider a commitment attempt as a request made by a debtor to push a creditor to be committed. Thus, when an agent Ag_1 requests another agent Ag_2 to do something, we say that the first agent is trying to induce the other agent to make a commitment. A commitment attempt is thought of as a type of social commitment because it conveys content which is made public once the attempt is performed. However, in our approach, there is a true commitment only after the creditor agent reacts in response to the commitment attempt. We distinguish four types of commitment attempts: propositional commitment attempts (***PCT***), action commitment attempts (***ACT***), conditional commitment attempts about propossitions (***CCTP***), and conditional commitment attempts about actions (***CCTA***).

Figure 7.1 illustrates the taxonomy explained in this section.

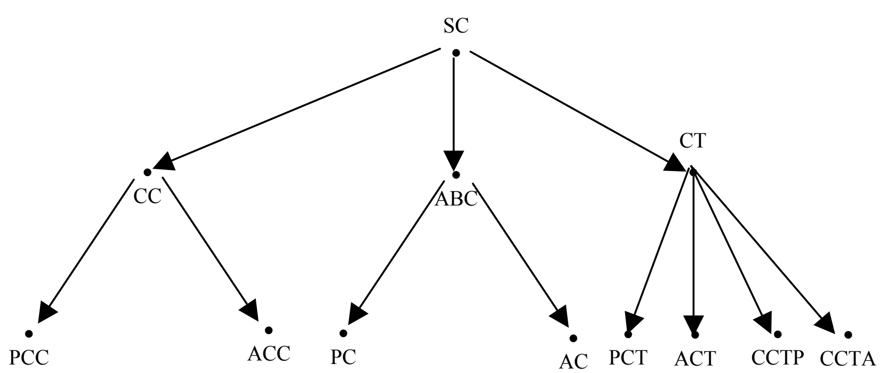


Figure 7.1. Social commitment taxonomy

In our framework, there is no explicit relation between propositional commitments and action commitments. When the current state of the world does not satisfy a propositional commitment, we speak about a violation of this commitment. There is no rule indicating that an agent develops an action commitment to make the content of its propositional commitment true when this commitment becomes violated. A propositional commitment is a commitment about a state of the world that the debtor agent can not realize. In contrast, an action commitment is a commitment about an action that the debtor commits to perform in the present or in the future.

In the two following sections we define the logical model (syntax and semantics) of our commitment and argument based-approach (CAN). We call this logical model $DCTL*_{CAN}$ because it is based on CTL* and Dynamic Logic.

7.3 Syntax

In this section we specify the syntax of the different elements that we use in our framework. These elements are: propositional elements, actions, social commitments, actions applied to commitments and argumentation relations.

Our formal language £ (the metalanguage) is based on an extended version of CTL* (Emerson and Halpern, 1986), (Hafer and Thomas, 1987) and on dynamic logic (Harel, 1979). Temporal logic and dynamic logic are two powerful logics developed to specify and to prove properties of computational processes (Harel, 1984), (Pnueli, 1986). We use a branching time for the future and we suppose that the past is linear (Ben-Ari et al., 1983). Each node in the branching time model is represented by a state s_i and a time point t_j (Figure 7.2). We also suppose that time is discrete. In our model, temporal logic enables us to express all the temporal aspects related to the handling of commitments and arguments. On one hand, we use the branching time in order to formalize the different choices that agents have when they participate in conversations. On the other hand, dynamic logic allows us to capture the actions that agents are committed to perform and the actions that agents perform on different commitments and commitment contents when they participate in these conversations. Indeed, from a philosophical point of view, action and branching time are logically related (Belnap, 1991). The actions of agents are not fully determined. Moreover, these actions can have many different possible future effects. For this reason, it is preferable to work out a logic of action that is compatible with indeterminism. According to indeterminism, several moments of time might follow the same moment in the future of the world. Any moment of time can belong to several paths (or *histories*) representing possible courses of the world with the same past and present but different historic continuations of that moment.

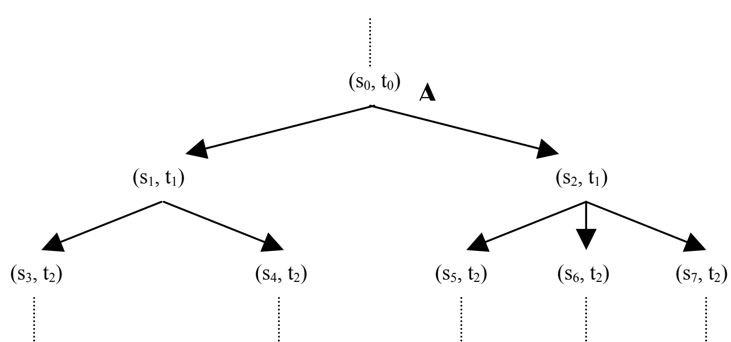


Figure 7.2. The branching time model

Let Φp be the set of atomic propositions and Φa be the set of atomic action symbols. The set of agents is denoted A and the set of time points is denoted TP. The various types of commitments, the agents' actions on commitments and on their contents and the argumentation relations are introduced as modal operators. We distinguish between *commitment formulae* and *commitment free formulae*. In this chapter, a commitment formula, independently of the commitment type, is denoted: $SC(Ag_1, Ag_2, t, \varphi)$ where t is the utterance time (time at which the commitment is created) and φ is a *commitment free formula*. A commitment free formula is a well-formed formula that does not have the form of a commitment formula. In a commitment formula Ag_1 and Ag_2 are two agents and φ is the commitment content. When t is unknown because the commitment is not yet created, we drop it from the commitment formula. In this case a commitment is denoted: $SC(Ag_1, Ag_2, , \varphi)$. In this logical model we use the symbol $\wedge$ in the object language and the symbol & in the metalanguage for "and". For "or" we use the symbol $\vee$ in the object language and the symbol | in the metalanguage. For "not" we use the same symbol $\neg$ in the two languages.

The language £ can be defined by the following syntactic rules.

7.3.1 Propositional Elements

Atomic formula
R1. $\forall\psi \in \Phi p, \psi \in £$.

Conjunction
R2. $p, q \in £ \Rightarrow p \wedge q \in £$.

Negation
R3. $p \in £ \Rightarrow \neg p \in £$.

Argumentation
R4. $p, q \in £ \Rightarrow p \therefore q \in £$.

This means that p is an argument for q. We can read this formula: *p*, *so q*. The property of *nonmonotonicity* of arguments does not appear at this level. The reason is that R4 introduces only argumentation as a logical relation between propositions. As Prakken and Vreeswijk argued, argumentation systems are able to incorporate the monotonic notions of logical consequence as a special case in their definition of what an argument is (Prakken and Vreeswijk, 2000). In our model, we capture the property of nonmonotonicity by the argumentation relations (attack, defense, justification, etc.). We deal with this aspect in the following sections.

Universal path-quantifier
R5. $p \in £ \Rightarrow Ap \in £$.

Existential path-quantifier
R6. $p \in £ \Rightarrow Ep \in £$.

***Until* (*in the future*)**
R7. $p, q \in £ \Rightarrow p\ U^+\ q \in £$.
Informally, $p\ U^+\ q$ (*p until q*) means that on a given path from the given moment, there is some future moment in which *q* will eventually hold and *p* holds at all moments until that future moment.

***Next moment* (*in the future*)**
R8. $p \in £ \Rightarrow X^+p \in £$.

X^+p holds at the current moment, if *p* holds at the next moment.

***Since* (*in the past*)**
R9. $p, q \in £ \Rightarrow p\ U^-\ q \in £$.

The intuitive interpretation of $p\ U^-\ q$ (*p since q*) is that on a given path from the given moment, there is some past moment in which *q* eventually held and *p* holds at all moments since that past moment.

***Previous moment* (*in the past*)**
R10. $p \in £ \Rightarrow X^-p \in £$.

X^-p holds at the current moment, if *p* held at the previous moment.

7.3.2 Actions

Action performance
R11. $p \in £ \ \&\ \alpha \in \Phi a \Rightarrow Perform(\alpha)p \in £$, where *p* is a commitment free formula.

Perform(α)*p* is an operator from dynamic logic. It indicates that the achievement of action α makes the proposition *p* true. This operator allows us to represent the fact that by way of performing actions, agents bring about facts in the world. They make true propositions representing these facts (Chellas, 1992).

7.3.3 Social Commitments

Propositional commitments
R12. $p \in £ \ \&\ t \in TP \ \&\ \{Ag_1, Ag_2\} \subseteq A \Rightarrow PC(Ag_1, Ag_2, t, p) \in £$, where *p* is a commitment free formula.

Action commitments
R13. $\alpha \in \Phi a \ \&\ p \in £ \ \&\ t \in TP \ \&\ \{Ag_1, Ag_2\} \subseteq A \Rightarrow AC(Ag_1, Ag_2, t, (\alpha, p)) \in £$, where *p* is a commitment free formula.

Conditional commitments about propositions
R14. $p, p' \in £ \ \&\ t \in TP \ \&\ \{Ag_1, Ag_2\} \subseteq A \Rightarrow PCC(Ag_1, Ag_2, t, (p, p')) \in £$, where *p* and *p'* are commitment free formulae.

Conditional commitments about actions
R15. $\alpha \in \Phi a$ & p, p' $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$
ACC(Ag_1, Ag_2, t, (p, (α, p'))) $\in$ £, where *p* and *p'* are commitment free formulae.

Commitment attempts
In order to formally introduce the notion of commitment attempt (syntax and semantics) we introduce the following definition.

Definition
$some(x, \{c_1, \ldots, c_n\}, p(x)) =_{def} p(c_1) \vee \ldots \vee p(c_n)$
where $c_1, \ldots, c_n$ are constant terms. A constant term can be a number, a name, etc.

We can define the syntax of propositional commitment attempts, action commitment attempts, conditional commitment attempts about propositions and conditional commitment attempts about actions as follows:

Propositional commitment attempts
R16. p $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$
PCT(Ag_1, Ag_2, t, some(x, $\{c_1, ..., c_n\}$, p(x))) $\in$ £, where *p* is a commitment free formula.

Action commitment attempts
R17. $\alpha \in \Phi a$ & p $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A$
$\Rightarrow$ ACT(Ag_1, Ag_2, t, (α, p)) $\in$ £, where *p* is a commitment free formula.

Conditional commitment attempts about propositions
R18. p, p' $\in$ £ & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$
CCTP(Ag_1, Ag_2, t, (p, some(x, $\{c_1, ..., c_n\}$, p'(x)))) $\in$ £, where *p* and *p'* are commitment free formulae.

Conditional commitment attempts about actions
R19. $\alpha \in \Phi a$ & p, p' $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A$
$\Rightarrow$ CCTA(Ag_1, Ag_2, t, (p, (α, p'))) $\in$ £, where *p* and *p'* are commitment free formulae.

Agent's desire about a propositional commitment from the addressee
R20. p $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$ Want(Ag_1, PC(Ag_2, Ag_1, t, p)) $\in$ £, where *p* is a commitment free formula.

This formula means that agent Ag_1 wants that agent Ag_2 commits that *p* is true.

Agent's desire about an action commitment from the addressee
R21. $\alpha \in \Phi a$ & p $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$
Want(Ag_1, AC(Ag_2, Ag_1, t, (α, p))) $\in$ £, where *p* is a commitment free formula.

Agent's desire about a propositional conditional commitment from the addressee
R22. p, p' $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$ Want(Ag_1, PCC(Ag_2, Ag_1, t, (p, p'))) $\in$ £, where *p* and *p'* are commitment free formulae.

Agent's desire about an action conditional commitment from the addressee
R23. $\alpha \in \Phi a$ & p, p' $\in$ £ & t $\in$ TP & $\{Ag_1, Ag_2\} \subseteq A \Rightarrow$
Want(Ag_1, ACC(Ag_2, Ag_1, t, (p, (α, p')))) $\in$ £, where *p* and *p'* are commitment free formulae.

7.3.4 Action Occurrences applied to Commitments

We use the abbreviation $SC(Ag_1, Ag_2, t, \varphi)$, where φ is a commitment free formula, to indicate a social commitment. The syntactical form of the commitment content φ depends of the commitment type according to the following rules:

If SC is a PC then φ has the syntactical form of *p*.
If SC is an AC then φ has the syntactical form of (α, p).
If SC is a PCC then φ has the syntactical form of (p, p').
If SC is an ACC then φ has the syntactical form of $(p, (\alpha, p'))$.
If SC is a PCT then φ has the syntactical form of $some(x, \{c_1, \ldots, c_n\}, p(x))$.
If SC is an ACT then φ has the syntactical form of (α, p).
If SC is a CCTP then φ has the syntactical form of $(p, some(x, \{c_1, \ldots, c_n\}, p'(x)))$.
If SC is a CCTA then φ has the syntactical form of $(p, (\alpha, p'))$.

Creation of a commitment
R24. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Create(Ag_1, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Withdrawal of a commitment
R25. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Withdraw(Ag_1, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Satisfaction (or fulfillment) of a commitment
R26. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Satisfy(Ag_1, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Violation of a commitment
R27. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Violate(Ag_1, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Reactivation of a commitment
R28. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Reactivate(Ag_1, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

An active commitment
R29. SC(Ag_1, Ag_2, φ) $\in$ £ $\Rightarrow$ Active(SC(Ag_1, Ag_2, φ)) $\in$ £.

7.3.5 Action Occurrences applied to Commitment Contents

Acceptation of a commitment content
R30. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Accept-content(Ag_2, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Refusal of a commitment content
R31. SC(Ag_1, Ag_2, t, φ) $\in$ £ $\Rightarrow$ Refuse-content(Ag_2, SC(Ag_1, Ag_2, t, φ)) $\in$ £.

Challenge of a commitment content
R32. $SC(Ag_1, Ag_2, t, \varphi) \in £ \Rightarrow$ Challenge-content$(Ag_2, SC(Ag_1, Ag_2, t, \varphi)) \in £$.

7.3.6 Argumentation Relations

In the argument-based approach, nonmonotonic, or defeasible reasoning is formalized by using notions like attack, defeat, defense and justification. Indeed, the property of nonmonotonicity is captured by the interaction of arguments for and against certain conclusions (Prakken and Vreeswijk, 2000). In this section, we introduce five argumentation relations in order to capture this property in our logical model.

Attack of a commitment content
R33 (1). $PC(Ag_1, Ag_2, t, p) \in £$ & $p' \in £ \Rightarrow$
Attack-content$(Ag_2, PC(Ag_1, Ag_2, t, p), p') \in £$, where *p'* is a commitment free formula.

We overload this formula as follows:

R33 (2). $PC(Ag_1, Ag_2, t, p) \in £ \Rightarrow$ Attack-content$(Ag_2, PC(Ag_1, Ag_2, t, p)) \in £$.

Defense of a commitment content against an attacker
R34 (1). $PC(Ag_1, Ag_2, t, p) \in £$ & $p' \in £ \Rightarrow$
Defend-content$(Ag_1, PC(Ag_1, Ag_2, t, p), p') \in £$, where *p'* is a commitment free formula.

R34(2). $PC(Ag_1, Ag_2, t, p) \in £ \Rightarrow$ Defend-content$(Ag_1, PC(Ag_1, Ag_2, t, p)) \in £$.

***Defense of a commitment content against all the attackers* (*strong defense*)**
R35 (1). $PC(Ag_1, Ag_2, t, p) \in £$ & $p' \in £ \Rightarrow$
Defend$^+$-content$(Ag_1, PC(Ag_1, Ag_2, t, p), p') \in £$, where *p'* is a commitment free formula.

R35 (2). $PC(Ag_1, Ag_2, t, p) \in £ \Rightarrow$ Defend$^+$-content$(Ag_1, PC(Ag_1, Ag_2, t, p)) \in £$.

Justification of a commitment content
R36 (1). $PC(Ag_1, Ag_2, t, p) \in £$ & $p' \in £ \Rightarrow$
Justify-content$(Ag_1, PC(Ag_1, Ag_2, t, p), p') \in £$, where *p'* is a commitment free formula.
R36 (2). $PC(Ag_1, Ag_2, t, p) \in £ \Rightarrow$ Justify-content$(Ag_1, SC(Ag_1, Ag_2, t, p)) \in £$.

Contradiction of a commitment content
R37. $PC(Ag_1, Ag_2, t, p) \in £sc \Rightarrow$ Contradict-content$(Ag_1, SC(Ag_1, Ag_2, t, p)) \in £sc$.

This relation means that an agent contradicts the content of its commitment.

Agent's desire about the justification of a commitment content from the addressee
R38. Justify-content(Ag_2, PC(Ag_2, Ag_1, t, p)) $\in$ £ $\Rightarrow$
Want(Ag_1, Justify-content(Ag_2, PC(Ag_2, Ag_1, t, p))) $\in$ £.

7.3.7 State and Path Formulae

As in CTL*, we have in our model two types of well-formed formulae : *state formulae* and *path formulae* (Emerson, 1990). State formulae are formulae which are evaluated (true or false) in particular states. Path formulae are formulae which are evaluated along certain paths.

R39. Any atomic formula is a state formula.

R40. Any state formula is a path formula.

R41. If p, q are state formulae, then $p \wedge q$, $\neg p$ are also state formulae.

R42. If p, q are path formulae, then $p \wedge q$, $\neg p$ are also path formulae.

R43. If p, q are path formulae, then $p\ U^+\ q$, X^+p, $p\ U^-\ q$, X^-p are also path formulae.

R44. If φ is a path formula, then SC(Ag_1, Ag_2, φ) is a state formula.

R45. Actions performed on commitments and on their contents:
if SC(Ag_1, Ag_2, φ) is a state formula, then
Act(Ag_1, SC(Ag_1, Ag_2, φ)) and Act-content(Ag_1, SC(Ag_1, Ag_2, φ)) are path formulae,
Want(Ag_1, SC(Ag_2, Ag_1, t, p))) and Want(Ag_1, Justify-content(Ag_2, PC(Ag_2, Ag_1, t, p))) are state formulae.

Abbreviations
We use in our model the following abbreviations:

A1. $p \vee q$ (disjunction) is the abbreviation of $\neg(\neg p \wedge \neg q)$

A2. $p \Rightarrow q$ (classical implication) is the abbreviation of $\neg p \vee q$

A3. true is the abbreviation of $p \vee \neg p$

A4. false is the abbreviation of $\neg$true

A5. F^+p (sometimes in the future) is the abbreviation of true U^+ p

A6. G^+p (globally in the future) is the abbreviation of $\neg F^+ \neg p$

A7. $F^{+\infty}$ (infinitely often in the future) is the abbreviation of G^+F^+p

A8. $G^{+\infty}$ (almost everywhere in the future) is the abbreviation of F^+G^+p

A9. p B^+ q (p before q in the future) is the abbreviation of $\neg((\neg p)\ U^+\ q)$

A10. F^-p (sometimes in the past) is the abbreviation of true U^- p

A11. G^-p (globally in the past) is the abbreviation of $\neg F^-\neg p$

A12. $F^{-\infty}$ (infinitely often in the past) is the abbreviation of G^-F^-p

A13. $G^{-\infty}$ (almost everywhere in the past) is the abbreviation of F^-G^-p

A14. p B^- q (p after q in the past) is the abbreviation of $\neg((\neg p)\ U^-\ q)$

7.4 Semantics

In this section, we define the formal model in which we evaluate the well-formed formulae of our framework. Thereafter, we give the semantics of the different elements that we specified syntactically in the previous section.

7.4.1 The Formal Model

Let *S* be a set of states and $R \subseteq S \times S$ be a transition relation indicating branching time. A path *Pa* is an infinite sequence of states $<s_0, s_1,\ldots>$ where: $\forall i \in \mathbb{N}, (s_i, s_{i+1}) \in R$ and $T(s_{i+1}) = T(s_i) + 1$. The function *T* gives us for each state s_i the corresponding moment *t* (this function will be specified later).

We use the notation s_i [*Pa* to indicate that the state s_i belongs to the path *Pa* (i.e. s_i appears in the sequence $<s_0, s_1,\ldots>$ that describes the path *Pa*). We denote the set of all paths by σ. The set of all paths traversing the state s_i are denoted: σ^{si}. We suppose that all paths start from s_0 ($T(s_0) = 0$).

In our vision of branching future, we can have several states at the same moment. Thus, in Figure 7.2 we have two different states: s_1 and s_2 at the same time t_1. At moment t_2 we have the states s_3, s_4, s_5, s_6, s_7. Along a given path (for example the real path) there is one and only one state at one moment. Indeed, in our framework, s_i does not indicate (necessarily) the state at moment *i*. Therefore, it is necessary to specify the state *s* and the moment *t* i.e. a pair $(s, t) \in S \times TP$.

According to this formalization, we can use the notation: M, s_i, $T(s_i) \models \psi$ to indicate that ψ is satisfied in the model *M* at state s_i at moment $T(s_i)$. To simplify this notation, we will use in the rest of this chapter the following abreviation: M, $s_i \models \psi$. In this notation: M, $s_i \models \psi$ there is a "hidden" time.

A formal model for £ is defined as follows:

M(S, R, A, TP, Np, Fap, T, Rsc, Rw)

where:

S : a nonempty set of states.

R : $R \subseteq S \times S$ a transition relation that defines all the transitions of the model.

A : a nonempty set of agents.

TP : a nonempty set of time points.

Np : S $\rightarrow$ $\mathbf{2^{\Phi p}}$: function relating each state $s \in S$ to the set of the atomic propositions that are true in this state.

Fap : S $\times$ **Φa** $\rightarrow$ $\mathbf{2^S}$: function that gives us the state transitions caused by the achievement of an action. For instance, in the Figure 7.3 we have : $Fap(s_i, \alpha) = \{s_j, s_l\}$. The transitions defined by *Fap* are a sub-set of the transitions defined by *R*. This function allows us to represent what is known in philosophical logic by "moments of time that are related by virtue of the actions of the agents". As Chellas pointed out (1992), to each moment *m* there corresponds the set of alternative moments *m'* which are compatible with all the actions that an agent *Ag* performs at moment *m*. These moments *m'* as under the control of, or responsive to the actions of, agent *Ag* at the moment *m*.

T : S $\rightarrow$ **TP** : function associating to any state s_i the corresponding time. For instance, in Figure 7.2 we have: $T(s_5) = t_2$.

Rsc : A $\times$ **A** $\times$ **S** $\rightarrow$ $\wp(\sigma)$: function producing the accessibility modal relations for social commitments. $\wp(\sigma)$ is a powerset of paths.

Rw : A $\times$ **S** $\rightarrow$ $\wp(\sigma)$: function producing the accessibility modal relations for agent' desires about the commitments of the addressee.

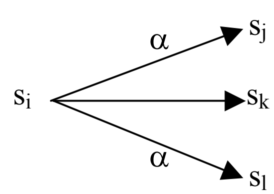


Figure 7.3. State transitions caused by the achievement of the action α

The function *Rsc* gives us all the paths along which the commitment created by an agent Ag_1 towards another agent Ag_2 *must be satisfied* (fulfilled). These paths are conceived as merely "possible", and as paths when the content of a commitment *should* be true. Indeed, the outputs of the function *Rsc* are known only after the creation of the commitments. Thus, this depends on the state in which the commitment is created. For example, if we have:

$Pa \in Rsc(Ag_1, Ag_2, s_i)$, then this means that at moment $T(s_i)$ agent Ag_1 is committed towards agent Ag_2 to satisfy a certain commitment along the path Pa. We can see that Rsc depends on the current moment $T(s_i)$.

As operators, the social commitments we introduced in our model and whose semantics will be defined on the basis of this relation are modal operators like the operator ($\Box$) (Chellas, 1980). The reading of $\Box p$ is as follows: an agent Ag_1 commits towards an agent Ag_2 that p is true or an action will be performed making p true.

The function $Rw(Ag_1, s_i)$ gives us the paths along which Ag_1 wants that the addressee commits or justifies its commitment. This accessibility modal relation will be used to define the semantics of the commitment attempts and the challenge of a commitment attempt.

Our logical model of absolute and conditional commitments is a KD modal logic (D: serial). This logic allows us to capture interesting intuitions about the manipulation of commitments. The *rule of necessitation* in this model can be expressed as follows: if p is a theorem, then $SC(Ag_1, Ag_2, t, p)$ is also a theorem. A commitment is a theorem iff it is satisfied in all states of the model. Semantically speaking, if the commitment-content is a theorem, then the commitment is always satisfied. However, expressed in such a way, this rule indicates that agents commit about all theorems. In the context of agent communication that we address in this thesis, this rule should be expressed as follows: if p is a theorem and an agent Ag_1 creates at moment t a commitment towards another agent Ag_2 about p, then $SC(Ag_1, Ag_2, t, p)$ is a theorem. In addition, the *N axiom* can be expressed as follows: if an agent commits towards another agent about a proposition, then it commits that this proposition is true or false (i.e. $PC(Ag_1, Ag_2, (p \vee \neg p))$).

The accessibility modal relation Rsc is *serial*, i.e.:

$$\forall Ag_1, Ag_2 \in A \ \& \ \forall s_i \in S \ \exists Pa \in \sigma : Pa \in Rsc(Ag_1, Ag_2, s_i)$$

This property fits with the notion of infinite paths in CTL*. It means that if an agent commits towards another agent that a proposition is true or that an action will be performed, then this agent does not commit about the negation of this proposition or so that this action will not be performed (i.e. $\Box p \Rightarrow \neg\Box\neg p$). An agent cannot commit about some thing and its negation.

The accessibility modal relation Rw is *serial*:

$$\forall Ag_1 \in A \ \& \ \forall s_i \in S \ \exists Pa \in \sigma : Pa \in Rw(Ag_1, s_i)$$

Therefore, the logic of commitment attempts is a KD modal logic.

As in CTL*, we have in our model path formulae and state formulae. We propose to evaluate the different types of commitments as state formulae. These formulae can also be interpreted on paths in which case one considers satisfaction in the first state of a path. On the other hand, we propose to evaluate the actions on commitments and the argumentation

relations on paths. These path formulae can be interpreted on states if they are true on all the paths traversing a given state. The notation M, $s_i \models \psi$ indicates that the formula ψ is evaluated in the state s_i of the model M. The notation M, Pa, $s_i \models \psi$ indicates that the formula ψ is evaluated at the state s_i along the path Pa where s_i [Pa.
We can now define the semantics of the elements of £ in the model M.

7.4.2 Propositional Elements

Atomic formula
S1. $M, s_i \models \psi$ iff $\psi \in Np(s_i)$ with $\psi \in \Phi p$

Conjunction
S2. $M, s_i \models p \wedge q$ iff $M, s_i \models p$ & $M, s_i \models q$

Negation
S3. $M, s_i \models \neg p$ iff $M, s_i \not\models p$

Argumentation
S4. $M, s_i \models p \therefore q$ iff $M, s_i \models p$ & $(\forall M' \in \mathcal{M}$ & $\forall s_j \in S_{M'}$ $M', s_j \models p \Rightarrow M', s_j \models q)$
where $\mathcal{M}$ is the set of models, and $S_{M'}$ is the set of states of the model M'.

We add the first clause to capture the following aspect: when an agent presents an argument p for q (i.e. $p \therefore q$) for this agent p is true and if p is true then q is true.

Universal path-quantifier
S5. $M, s_i \models$ Ap iff $(\forall Pa\ Pa \in \sigma^{si} \Rightarrow M, Pa, s_i \models p)$

Existential path-quantifier
S6. $M, s_i \models$ Ep iff $(\exists Pa \in \sigma^{si}$ & $M, Pa, s_i \models p)$

Propositional path formulae
S7. $M, Pa, s_i \models \psi$ iff $M, s_i \models \psi$ with $\psi \in \Phi p$

S8. $M, Pa, s_i \models p \wedge q$ iff $M, Pa, s_i \models p$ & $M, Pa, s_i \models q$

S9. $M, Pa, s_i \models \neg p$ iff $M, Pa, s_i \not\models p$

S10. $M, Pa, s_i \models p \therefore q$ iff
$M, Pa, s_i \models p$ & $(\forall M' \in \mathcal{M}$ & $\forall s_j \in S_{M'}$ & $\forall Pa' \in \sigma_{M'} : s_j$ [Pa'
$M', Pa', s_j \models p \Rightarrow M', Pa', s_j \models q)$
where $\mathcal{M}$ is the set of models, $S_{M'}$ is the set of states of the model M', and $\sigma_{M'}$ is the set of paths of the model M'.

Until (in the future)
S11. $M, Pa, s_i \models p\ U^+\ q$ iff $(\exists s_j : s_j$ [Pa & $T(s_i) \leq T(s_j)$ & $M, Pa, s_j \models q$

$\& (\forall s_k\ T(s_i) \leq T(s_k) < T(s_j)\ \ \&\ s_k\ [\ Pa \Rightarrow M, Pa, s_k \vDash p))$

***Next moment* (*in the future*)**

S12. $M, Pa, s_i \vDash X^+p$ iff $M, Pa, s_j \vDash p$ where $T(s_j) = T(s_i) + 1\ \&\ s_j\ [\ Pa$

***Since* (*in the past*)**

S13. $M, Pa, s_i \vDash p\ U^-\ q$ iff $(\exists s_j : s_j\ [\ Pa\ \&\ T(s_j) \leq T(s_i)\ \&\ M, Pa, s_j \vDash q$
$\& (\forall s_k\ T(s_j) < T(s_k) \leq T(s_i)\ \&\ s_k\ [\ Pa \Rightarrow M, Pa, s_k \vDash p))$

***Previous moment* (*in the past*)**

S14. $M, Pa, s_i \vDash X^-p$ iff $M, Pa, s_j \vDash p$ where $T(s_j) = T(s_i) - 1\ \&\ s_j\ [\ Pa$

For more clearness, we give the semantics of some abbreviations that we consider as propositions (P15-P25)

Sometimes in the future

P15. $M, Pa, s_i \vDash F^+p$ iff $\exists s_j : s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i)\ \&\ M, Pa, s_j \vDash p$

Globally in the future

P16. $M, Pa, s_i \vDash G^+p$ iff $\forall s_j\ s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i) \Rightarrow M, Pa, s_j \vDash p$

Infinitely often in the future

P17. $M, Pa, s_i \vDash F^{+\infty}p$ iff $\forall s_j\ s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i) \Rightarrow M, Pa, s_j \vDash F^+p$

In other words

P18. $M, Pa, s_i \vDash F^{+\infty}p$ iff $\forall s_j\ s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i) \Rightarrow \exists s_k : (s_k\ [\ Pa\ \&\ T(s_k) \geq T(s_j)$
$\&\ M, Pa, s_k \vDash p)$

Almost everywhere in the future

P19. $M, Pa, s_i \vDash G^{+\infty}p$ iff $\exists s_j : s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i)\ \&\ (\forall s_k\ s_k\ [\ pa\ \&\ T(s_k) \geq T(s_j)$
$\Rightarrow M, Pa, s_k \vDash p)$

p before q in the future

P20. $M, Pa, s_i \vDash p\ B^+\ q$ iff $\forall s_j\ (s_j\ [\ Pa\ \&\ T(s_j) \geq T(s_i)\ \&\ M, Pa, s_j \vDash q)$
$\Rightarrow (\exists s_k : s_k\ [\ Pa\ \&\ T(s_i) \leq T(s_k) < T(s_j)\ \&\ M, Pa, s_k \vDash p)$

Sometimes in the past

P21. $M, Pa, s_i \vDash F^-p$ iff $\exists s_j : s_j\ [\ Pa\ \&\ T(s_j) \leq T(s_i)\ \&\ M, Pa, s_j \vDash p$

Globally in the past

P22. $M, Pa, s_i \vDash G^-p$ iff $\forall s_j\ s_j\ [\ Pa\ \&\ T(s_j) \leq T(s_i) \Rightarrow M, Pa, s_j \vDash p$

Infinitely often in the past

P23. $M, Pa, s_i \vDash F^{-\infty}p$ iff $\forall s_j\ s_j\ [\ Pa\ \&\ T(s_j) \leq T(s_i) \Rightarrow \exists s_k : (s_k\ [\ Pa\ \&\ T(s_k) \leq T(s_j)$

$\& M, Pa, s_k \vDash p)$

Almost everywhere in the past

P24. $M, Pa, s_i \vDash G^{-\infty}p$ iff $\exists s_j : s_j [Pa \& T(s_j) \leq T(s_i) \& (\forall s_k\ s_k [Pa \& T(s_k) \leq T(s_j) \Rightarrow M, Pa, s_k \vDash p)$

p before q in the past

P25. $M, Pa, s_i \vDash p\ B^{-}\ q$ iff $\forall s_j\ (s_j [Pa \& T(s_j) \leq T(s_i) \& M, Pa, s_j \vDash q) \Rightarrow (\exists s_k : s_k [Pa \& T(s_j) < T(s_k) \leq T(s_j) \& M, Pa, s_k \vDash p)$

In the following sections we specify our semantics in the form of definitions and properties that follow from these definitions.

7.4.3 Actions

In this section we give the semantics of action performance. This semantics is expressed by using *Perform*(α)*p* operator.

Definition

Action performance

S26. $M, Pa, s_i \vDash Perform(\alpha)p$ iff $Fap(s_i, \alpha) \neq \varnothing \& \forall s_j\ s_j \in Fap(s_i, \alpha) \& s_j [Pa \Rightarrow M, Pa, s_j \vDash p$

The fact that $Fap(s_i, \alpha) \neq \varnothing$ means that *Perform*(α)*p* is actual and not conditional.

S27. $M, s_i \vDash Perform(\alpha)p$ iff $\forall Pa\ Pa \in \sigma^{si} \Rightarrow M, Pa, s_i \vDash Perform(\alpha)p$

7.4.4 Social Commitments

In this section we define the semantics of different types of social commitments according to the taxonomy that we specified in Section 7.2.

Definitions

Social commitment as a path formula

S28. $M, Pa, s_i \vDash SC(Ag_1, Ag_2, t, \varphi)$ iff $M, s_i \vDash SC(Ag_1, Ag_2, t, \varphi)$

Propositional commitments

S29. $M, s_i \vDash PC(Ag_1, Ag_2, t, p)$ iff $\forall Pa\ Pa \in Rsc(Ag_1, Ag_2, s_i) \Rightarrow \exists s_j [Pa : T(s_j) = T(s_i) \& M, Pa, s_j \vDash p$

We notice here that we evaluate *p* along an accessible path *Pa* at a state s_j that can be different from the current state s_i. This allows us to model agents' uncertainty about this current state. This means that we do not assume that agents know the current state. However, we assume that these agents know which time is associated to each state.

This formula gives us the semantics of propositional commitments in terms of accessible paths. The commitment is satisfied in a model at a state s_i iff its content is satisfied in the model along all accessible paths. This formula gives us the meaning of a social commitment, but states nothing about the fact that the agent must commit that some thing is true. Consequently, the omniscience problem in the sense that the agent commits that all the theorems are true is not present in our logic. On the other hand, to capture the idea that the agent commits that some proposition is true, we use dynamic logic.

Action commitments

S30. $AC(Ag_1, Ag_2, t, (\alpha, p)) =_{def} PC(Ag_1, Ag_2, t, Perform(\alpha)p)$

The formula S29 indicates that the commitment of agent Ag_1 towards agent Ag_2 about a proposition p is satisfied in the model iff along all accessible paths Pa p is true. The formula S30 indicates that agent Ag_1 is committed towards agent Ag_2 to do α and that along all accessible paths Pa performing α makes p true. According to formulae S29 and S30, the semantics we give to the commitments requires their fulfillment. Thus, if it is created, a commitment must be held. This satisfaction-based semantics reflects the idea of "prior possible choices of agents" that Belnap and Perloff used in their logic of agency (Belnap and Perloff, 1992). In this logic, agents make choices in time. In our model, these choices are represented by the commitments created by these agents. The notion of acting or choosing at a moment m is thought of in Belnap and Perloff's logic as constraining the course of events to lie within some particular subset of the possible histories available at that moment. This subset of the possible histories is represented by the set of paths along which the commitment must be satisfied. However, it is always possible to violate or withdraw such a commitment. For this reason, these two operations (violation and withdrawal) are explicitly included in our framework. Thus, it is possible to have wrong commitments because the accessibility relation Rsc gives us the paths along which the commitment created by an agent Ag_1 towards another agent Ag_2 must be satisfied.

Conditional commitments about propositions

S31. $M, s_i \vDash PCC(Ag_1, Ag_2, t, (p, p')) \text{ iff } (\exists Pa \in \sigma^{si} \ \& \ \exists s_j \, [\, Pa : T(s_j) \geq T(s_i) \ \& \ M, s_j \vDash p)$
$\Rightarrow M, s_j \vDash PC(Ag_1, Ag_2, t, p')$

This formula indicates that agent Ag_1 commits that p is true only if the condition p is true (or is satisfied).

Conditional commitment about actions

S32. $ACC(Ag_1, Ag_2, t, (p, (\alpha, p'))) =_{def} PCC(Ag_1, Ag_2, t, (p, Perform(\alpha)p'))$

Agent's desire about a propositional commitment from the addressee

S33. $M, s_i \vDash Want(Ag_1, PC(Ag_2, Ag_1, , p)) \text{ iff } \forall Pa \ Pa \in Rw(Ag_1, s_i) \Rightarrow$
$\exists s_j \, [\, Pa : T(s_j) = T(s_i) \ \& \ M, Pa, s_j \vDash PC(Ag_2, Ag_1, , p))$

Ag_1's desire about a propositional commitment of Ag_2 whose content is p is satisfied in the model iff along all accessible paths via Rw, Ag_2 commits towards Ag_1 that p. In the same way we can define the semantics of an agent's desire about the other commitment types.

Propositional commitment attempts

S34. M, $s_i \models$ PCT(Ag_1, Ag_2, t, some(x, {c_1, ..., c_n}, p(x))) iff

M, $s_i \models$ PC(Ag_1, Ag_2, t, Want(Ag_1, PC(Ag_2, Ag_1, , p(c_1)) $\vee$ … $\vee$ PC(Ag_2, Ag_1, p(c_n))))

The Ag_1's propositional commitment attempt towards Ag_2 is satisfied in the model iff Ag_1 commits that it wants that Ag_2 commits at a certain moment that one of the propositions $p(c_i)$ is true. This notion of commitment attempt captures open and yes/no questions.

Action commitment attempts

S35. M, $s_i \models$ ACT(Ag_1, Ag_2, t, (α, p)) iff

M, $s_i \models$ PC(Ag_1, Ag_2, t, Want(Ag_1, AC(Ag_2, Ag_1, , (α, p))))

The Ag_1's action commitment attempt towards Ag_2 is satisfied in the model iff Ag_1 commits that it wants that Ag_2 commits to perform the action. In the same way we can define the semantics of conditional commitment attempts about propositions and about actions.

7.4.5 Actions applied to Commitments

In this section we specify the semantics of different actions that agents can apply on their commitments. These actions are: creation, withdrawal, satisfaction, violation and reactivation. We also specify the relation between satisfaction and violation and we discuss the link between commitment states and these different actions.

Definitions

Create a social commitment

S36. M, Pa, $s_i \models$ Create(Ag_1, SC(Ag_1, Ag_2, t, φ)) iff

$\exists\alpha \in \Phi a$ & M, Pa, $s_i \models$ Perform(α)SC(Ag_1, Ag_2, t, φ) & t = T(s_i)

This formula indicates that the creation of a commitment is satisfied in the model *M* along a path *Pa* iff there is an action α whose performance makes true the commitment (i.e. the commitment holds after the performance of the action α) and if the creation moment is equal to the time associated to the current state. This formula highlights the fact that the creation of a commitment is an action in itself. Indeed, the action α corresponds to the agent's utterance which creates the commitment.

Withdraw a social commitment

S37. M, Pa, $s_i \models$ Withdraw(Ag_1, SC(Ag_1, Ag_2, t, φ)) iff

$\exists\alpha \in \Phi a$ &

M, Pa, $s_i \models$ F^-Create(SC(Ag_1, Ag_2, t, φ)

$\wedge$ ($\neg F^-$ Satisfy(Ag_1, PC(Ag_1, Ag_2, t, φ)))

$\wedge$ ($\neg F^-$ Violate(Ag_1, PC(Ag_1, Ag_2, t, φ)))

$\wedge$ Perform(α)$\neg$ SC(Ag_1, Ag_2, t, φ))

This formula indicates that an agent withdraws its commitment for φ iff the following conditions are satisfied:

1- The agent has already created this commitment in the past.
2- The commitment is not yet satidfied or violated in the past
3- The agent performs an action α so that this commitment does not hold at the current moment.

In addition, we add the following meaning postulate which is a constraint that agents must respect when communicating.

Meaning postulate
M38. $AG^+(Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
X^-
$(\neg Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
U^-
$Create(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \vee Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$

According to this constraint, if an agent withdraws its commitment, this means that before the current moment this commitment is not withdrawn since its creation or last reactivation.

On the other hand, commitments are persistent until their withdrawal. Formally, we have the following meaning postulate:

Meaning postulate
M39. $AG^+(SC(Ag_1, Ag_2, t, \varphi) \wedge \neg Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$X^+ SC(Ag_1, Ag_2, t, \varphi))$

Satisfy a propositional commitment
S40. $M, Pa, s_i \vDash Satisfy(Ag_1, PC(Ag_1, Ag_2, t, p))$ iff
$\exists Pa' \in \sigma \ \& \ \exists s_j \ T(s_j) \leq T(s_i) \ \& \ M, Pa', s_j \vDash CreatePC(Ag_1, Ag_2, t, p))$
$\& \ Pa \in Rsc(Ag_1, Ag_2, s_j)$

A propositional commitment is satisfied along a path *Pa* at a state s_i iff it was already created, and the path *Pa* is accessible via the relation *Rsc*. This means that, the path *Pa* corresponds to the satisfaction path of the commitment which is true at the state s_j. Along this accessible path the commitment content is true.

In addition, we add the following meaning postulate indicating that globally in all paths, if a commitment is withdrawn and not reactivated in the future, globally it can not be satisfied or violated.

Meaning postulate
M41. $AG^+(Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \wedge \neg F^+ Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\Rightarrow G^+(\neg Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \wedge \neg Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$
Satisfy an action commitment
S42. $Satisfy(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, p))) =_{def} Satisfy(Ag_1, PC(Ag_1, Ag_2, t, Perform(\alpha)p))$

Satisfy a conditional commitment about a proposition

S43. $M, Pa, s_i \vDash Satisfy(Ag_1, PCC(Ag_1, Ag_2, t, (p, p')))$ iff
$M, Pa, s_i \vDash F^{-}p \wedge Satisfy(Ag_1, PC(Ag_1, Ag_2, t, p))$.

A conditional commitment is satisfied in the model *M* along the path *Pa* iff the underlying condition *p* is satisfied in the past and that the debtor satisfies in (M, Pa, s_i) the resulting commitment $PC(Ag_1, Ag_2, t, p)$. In the same way we define the semantics of a conditional commitment about an action.

Satisfy a conditional commitment about an action

S44. $Satisfy(Ag_1, ACC(Ag_1, Ag_2, t, (p, (\alpha, p')))) =_{def}$
$Satisfy(Ag_1, PCC(Ag_1, Ag_2, t, (p, Perfoprm(\alpha)p')))$.

Satisfy a propositional commitment attempt

S45. $M, Pa, s_i \vDash Satisfy(Ag_2, PCT(Ag_1, Ag_2, t, some(x, \{c_1,..., c_n\}, p(x))))$ iff
$M, Pa, s_i \vDash Satisfy(Ag_2, PC(Ag_2, Ag_1, , p(c_1)))$
$\vee \ldots \vee Satisfy(Ag_2, PC(Ag_2, Ag_1, , p(c_n)))$

A propositional commitment attempt is satisfied by the creditor iff this agent satisfies the resulting propositional commitment. In the same way we define the semantics of the satisfaction of the other commitment attempt types.

Satisfy an action commitment attempt

S46. $M, Pa, s_i \vDash Satisfy(Ag_2, ACT(Ag_1, Ag_2, t, (\alpha, p)))$ iff
$M, Pa, s_i \vDash Satisfy(Ag_2, AC(Ag_2, Ag_1, , (\alpha, p)))$

Satisfy a conditional commitment attempt about a proposition

S47. $M, Pa, s_i \vDash Satisfy(Ag_2, CCTP(Ag_1, Ag_2, t, (p, some(x, \{c_1, ..., c_n\}, p'(x)))))$ iff
$M, Pa, s_i \vDash F^{-}p \wedge Satisfy(Ag_2, PCT(Ag_1, Ag_2, , some(x, \{c_1,..., c_n\}, p'(x))))$

Satisfy a conditional commitment attempt about an action

S48. $M, Pa, s_i \vDash Satisfy(Ag_2, CCTA(Ag_1, Ag_2, t, (p, (\alpha, p'))))$ iff
$M, Pa, s_i \vDash F^{-}p \wedge Satisfy(Ag_2, ACT(Ag_1, Ag_2, , (\alpha, p')))$

In the same way, the violation of the different types of commitments can be formulated. We give here just the definition of the violation of a propositional commitment

Violate a propositional commitment

S49. $M, Pa, s_i \vDash Violate(Ag_1, PC(Ag_1, Ag_2, t, p))$ iff
$\exists s_j\ T(s_j) \leq T(s_i)\ \&\ M, s_j \vDash PC(Ag_1, Ag_2, t, p))$
$\&\ Pa \notin Rsc(Ag_1, Ag_2, s_j)$

A propositional commitment is violated along a path *Pa* at a state s_i iff it already exists, and the path *Pa* does not correspond to the satisfaction path of the commitment which is true at the state s_j. Along this path the commitment content is false.

We have also the following proposition:

Proposition

P50. $M, Pa, s_i \models Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$ iff

$\exists s_j\ T(s_j) \leq T(s_i)\ \&\ M, s_j \models SC(Ag_1, Ag_2, t, \varphi))$

$\&\ M, Pa, s_i \models \neg Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$

The proof is a consequence of the definitions.

Reactivate a social commitment

S51. $M, Pa, s_i \models Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$ iff

$\exists\alpha \in \Phi a\ \&$

$M, Pa, s_i \models X^{-}F^{-}Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$

$\wedge (\neg F^{-}\ Satisfy(Ag_1, PC(Ag_1, Ag_2, t, p)))$

$\wedge (\neg F^{-}\ Violate(Ag_1, PC(Ag_1, Ag_2, t, p)))$

$\wedge\ Perform(\alpha)SC(Ag_1, Ag_2, t, \varphi)$

A commitment is reactivated iff:

1- It was previously withdrawn.
2- The commitment is not yet satidfied or violated in the past
3- The agent performs an action making the commitment true at the current moment.

Like for withdrawl, we add the following meaning postulate which is a constrant that agents must respect when communicating.

Meaning postulate

C52. $AG^{+}(Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$

X^{-}

$(\neg Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$

U^{-}

$Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$

According to this constraint, if an agent reactivates its commitment, this means that before the current moment this commitment is not reactivated since its last withdrawal.

Commitment states

The semantics of the actions that agents apply to commitment contents is related to the notion of commitment states (see Chapter 5). Thus, the semantics of these actions must be defined in terms of the semantics of these commitment states. Since a commitment state only holds as a result of the debtor's action, the semantics of a commitment state is determined by the operation that leads to this state. For example, the operation "withdraw" leads to the state "withdrawn". The semantics of the actions applied on the commitment contents requires a combination of all possible commitment states. An agent cannot act on a commitment content whose state is withdrawn. Thus, to simplify the notation, we suppose that a commitment is either active, or not active (withdrawn).

After introducing the different actions that the debtor can apply to its commitment, we can define the semantics of an active commitment as follows:

S53. $M, Pa, s_i \vDash Active(SC(Ag_1, Ag_2, t, \varphi))$ iff
$$M, Pa, s_i \vDash \neg Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)))\ U^{-}\ (Create(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \vee Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi)))$$

This property indicates that a commitment is active iff the two following conditions are satisfied (see Figure 7.4):

1- This commitment was already created **or** reactivated.
2- Until the current moment, the commitment was not withdrawn.

Therefore, once the commitment is withdrawn, it becomes inactive.

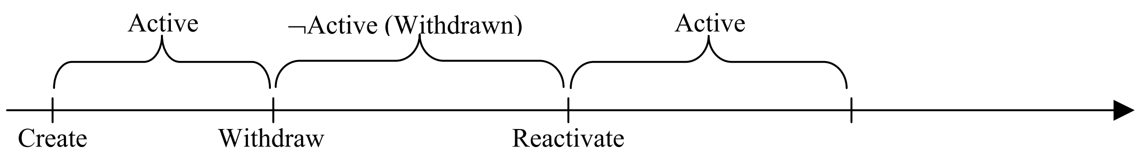


Figure 7.4. Social commitment activation

The formula S53 explains a persistence property of social commitments. A social commitment is persistent while it is active. This means that, it is persistent in all the states following the state in which it was created until its withdrawal, violation or satisfaction. The active state is satisfied in the model in the state in which the commitment is created and in all the states until its withdrawal, satisfaction or violation. In addition, we have the following properties:

Properties

P54. $AG^{+}\mathbf{(}Active(PC(Ag_1, Ag_2, t, p)) \wedge Active(PC(Ag_1, Ag_2, t', q) \Leftrightarrow Active(PC(Ag_1, Ag_2, , p \wedge q))\mathbf{)}$

P55. $AG^{+}\mathbf{(}Active(PC(Ag_1, Ag_2, t, p)) \vee Active(PC(Ag_1, Ag_2, t, q) \Leftrightarrow Active(PC(Ag_1, Ag_2, , p \vee q))\mathbf{)}$

The proof of $\Leftarrow$ is straightforward. The proof of $\Rightarrow$ is a consequence of the semantics of *Active*.

7.4.6 Actions applied to Commitment Contents

In this section we define the semantics of different actions that agents can perform on their commitment contents or on the commitment contents of other agents. These actions are: acceptation, refusal, and challenge.

Definitions

Accept a commitment content

S56. $M, Pa, s_i \models$ Accept-content(Ag_2, SC(Ag_1, Ag_2, t, φ)) iff :

$M, Pa, s_i \models$ Active(SC(Ag_1, Ag_2, t, φ)) $\wedge$ Create(Ag_2, SC(Ag_2, Ag_1, T(s_i), φ))

This formula indicates that the acceptance of the commitment content φ by agent Ag_2 is satisfied in the model M along a path Pa iff:

1- The commitment is active on this path because we cannot act on a commitment content if the commitment is not active.

2- Agent Ag_2 creates a commitment whose content is φ. Therefore, Ag_2 becomes committed towards the content φ.

Refuse a commitment content

S57. $M, Pa, s_i \models$ Refuse-content(Ag_2, SC(Ag_1, Ag_2, t, φ)) iff :

$M, Pa, s_i \models$Active(SC(Ag_1, Ag_2, t, φ)) & Create(Ag_2, SC(Ag_2, Ag_1, T(s_i), $\neg\varphi$))

The refusal of the commitment content φ by an agent Ag_2 is satisfied in the model M along a path Pa iff:

1- The commitment is active on this path.

2- Agent Ag_2 creates a commitment whose content is $\neg\varphi$. Therefore, Ag_2 becomes committed towards the content $\neg\varphi$.

Refusal is thus the dual notion of acceptance.

Challenge a commitment content

S58. $M, Pa, s_i \models$ Challenge-content(Ag_2, SC(Ag_1, Ag_2, t, φ)) iff

$\exists\alpha \in \Phi a$ &

$M, Pa, s_i \models$ Active(SC(Ag_1, Ag_2, t, φ))

$\wedge$ Perform(α)Want(Ag_2, Justify-content(Ag_1, SC(Ag_1, Ag_2, t, φ)))

This formula indicates that the challenge of the commitment content φ by an agent Ag_2 is satisfied in the model M along a path Pa iff:

1- The challenged commitment is active on this path.

2- Agent Ag_2 performs an action so that it wants that agent Ag_1 justifies its commitment content φ.

This formula highlights the fact that the challenge of a commitment content is an action in itself. As for the creation operation, the action α corresponds to the production of the utterance that challenges the commitment content.

7.4.7 Argumentation Relations

In this section we define the semantics of the argumentation relations that we introduced in Section 7.3.6. These argumentation relations are: justification, attack, defend, defend$^+$ and contradiction. We also formulate an interesting property that enables us to reflect the nonmonotonic nature of arguments.

Definition of basic notions

Justify the content of a social commitment

S59 (1). $M, Pa, s_i \vDash$ Justify-content(Ag_1, PC(Ag_1, Ag_2, t, p), p') iff

$M, Pa, s_i \vDash$ Active(PC(Ag_1, Ag_2, t, p)) & Create(Ag_1, PC(Ag_1, Ag_2, T(s_i), p' $\therefore$ p))

This formula indicates that the justification of the commitment content φ by an agent Ag_1 is satisfied in the model M on a path Pa iff:

1- This commitment is active on this path.

2- This agent creates on this path a commitment whose content is *p'* that supports the conclusion *p*.

In other words, a social commitment of an agent to another one to make a content *p* true is justified (by means of *p'*) iff the social commitment exists (has been created) and moreover a social commitment is created to establish an argument (*p'*, *p*), where *p'* is committed to be true because accordingly to the definition of the connector ($\therefore$), *p'* is true for Ag_1. The fact that this operator is included in the commitment indicates that the agent is committed that *p'* is true and then *p* is true, i.e. *p* is true because *p'* is true. We define the semantics of the overloaded formula of *Justify-content* as follows:

S59 (2). $M, Pa, s_i \vDash$ Justify-content(Ag_1, PC(Ag_1, Ag_2, t, p)) iff

$\exists p' \in £$:

$M, Pa, s_i \vDash$ Justify-content(Ag_1, PC(Ag_1, Ag_2, t, p), p')

We notice here that the purpose of this chapter is to give a semantics of the different actions that agents can perform when conversing. Thus, how do agents choose an argument among others and how do we ensure that the argumentation process terminates are questions that are addressed in Chapters 5 and 8.

The justification operation is the basis of other argumentation operations. As shown by the following definitions (formulae S54, S55, S56), this is due to the fact that all the other operations are defined using this operation.

Contradict the content of a social commitment

S60. $M, Pa, s_i \vDash$ Contradict-content(Ag_1, PC(Ag_1, Ag_2, t, p)) iff

$\exists p' \in £$:

$M, Pa, s_i \vDash$ Active(PC(Ag_1, Ag_2, t, p)) $\wedge$ Create(Ag_1, PC(Ag_1, Ag_2, T(si), p' $\therefore \neg$p))

This formula indicates that an agent contradicts its previous commitment whose content is *p* if it creates another commitment whose content is a logical conclusion of $\neg p$, whereas its commitment for *p* is still active.

Definition of derived notions

Attack the content of a social commitment

S61 (1). $M, Pa, s_i \models$ Attack-content(Ag_2, PC(Ag_1, Ag_2, t, p), p') iff
M, $Pa, s_i \models$ Active(PC(Ag_1, Ag_2, t, p))
$\wedge$ Justify-content(Ag_2, PC(Ag_2, Ag_1, T(s_i), $\neg$p), p')

This formula indicates that the attack of the commitment content *p* by an agent Ag_2 is satisfied in the model *M* along a path *Pa* iff:

1. This commitment is active on this path.
2. This agent justifies along this path its commitment whose content is $\neg p$.

S61 (2). $M, Pa, s_i \models$ Attack-content(Ag_2, PC(Ag_1, Ag_2, t, p)) iff
$\exists$p' $\in$ £ :
$M, Pa, s_i \models$ Attack-content(Ag_2, PC(Ag_1, Ag_2, t, p), p')

Defend the content of a social commitment

S62 (1). $M, Pa, s_i \models$ Defend-content(Ag_1, PC(Ag_1, Ag_2, t, p), p') iff
$\exists$p'' $\in$ £ :
$M, Pa, s_i \models$ Active(PC(Ag_1, Ag_2, t, p))
$\wedge$ X^-F^-Attack-content(Ag_2, PC(Ag_1, Ag_2, t, p), p''))
$\wedge$ Attack-content(Ag_1, SC(Ag_2, Ag_1, , p''), p'))

This formula indicates that the defense of the commitment content *p* by an agent Ag_1 is satisfied in the model *M* along a path *Pa* iff:

1. This commitment is active on this path.
2. This agent attacks the attacker of the content of its commitment.

S62 (2). $M, Pa, s_i \models$ Defend-content(Ag_1, PC(Ag_1, Ag_2, t, p)) iff
$\exists$p' $\in$ £ :
M, Pa, si $\models$ Defend-content(Ag_1, PC(Ag_1, Ag_2, t, p), p')

Defend strongly the content of a social commitment

S63. $M, Pa, s_i \models$ Defend^+-content(Ag_1, PC(Ag_1, Ag_2, t, p)) iff
$M, Pa, s_i \models$ Active(PC(Ag_1, Ag_2, t, p))
& ($\forall$p'' $\in$ £
$M, Pa, s_i \models$ X^-F^-Attack-content(Ag_2, PC(Ag_1, Ag_2, t, p), p'')
$\Rightarrow$ Attack-content(Ag_1, PC(Ag_2, Ag_1, , p'')))

This formula indicates that the strong defense of the commitment content φ by an agent Ag_1 is satisfied in the model *M* in along a path *Pa* iff:

1. This commitment is active on this path.

2. This agent attacks all the attackers of the content of its commitment.

Agent's desire about the justification of a commitment content from the addressee

S64. $M, s_i \vDash Want(Ag_1, Justify\text{-}content(Ag_2, PC(Ag_2, Ag_1, t, p)))$ iff

$\forall Pa\ Pa \in Rw(Ag_1, s_i) \Rightarrow$

$\exists s_j\ [\ Pa : T(s_j) = T(s_i)\ \&\ M, Pa, s_j \vDash F^{+}Justify\text{-}content(Ag_2, PC(Ag_2, Ag_1, t, p)))$

Ag_1's desire about the justification of a commitment of Ag_2 is satisfied in the model iff along all accessible paths via *Rw*, Ag_2 justifies in the future this commitment.

Property of nonmonotonicity

According to the property of nonmonotonicity, adding arguments can lead to the defeat of existing arguments. An argument is defeated if it is attacked successfully by a counterargument. In other words, an argument becomes invalid when it is attacked and it cannot be defended. In our model, that results in the following meaning postulate: in all paths of the model *M*, if Ag_2 attacks the content *p* of Ag_1's commitment and if Ag_1 cannot defend this content or attack the content of Ag_2's commitment, then Ag_1's commitment becomes unsatisfied in the model *M*. Formally, we have the following meaning postulate:

Meaning postulate

M65 (1). $AG^{+}\mathbf{(}Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, p), p')$

$\wedge\ G^{+}\mathbf{(}\neg Defend\text{-}content(Ag_1, SC(Ag_1, Ag_2, t, p))$

$\wedge\ \neg Attack\text{-}content(Ag_1, SC(Ag_2, Ag_1, , p'))\mathbf{)}$

$\Rightarrow \neg PC(Ag_1, Ag_2, t, p)\mathbf{)}$

In defeasible reasoning, an argument is valid until a counterargument attacks it. This property can be formally specified in our model by the following meaning postulate:

Meaning postulate

M65 (2). $AG^{+}\mathbf{(}Create(Ag_1, PC(Ag_1, Ag_2, t, p \therefore p')) \Rightarrow$

$X^{+}\mathbf{(}G^{+}(PC(Ag_1, Ag_2, t, p \therefore p'))$

$\vee\ \mathbf{(}PC(Ag_1, Ag_2, t, p \therefore p')$

$U^{+}\ Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, p \therefore p'))\mathbf{)))}$

This property indicates that in all paths of the model if a commitment whose content is an argument is created, then in the next state this commitment is either globally valid or it is valid until a counterargument attacks it. This property can be formulated using the *week until operator* U^{+w} of CTL* as follows:

M65 (3). $AG^{+}\ \mathbf{(}Create(Ag_1, PC(Ag_1, Ag_2, t, p \therefore p')) \Rightarrow$

$X^{+}\mathbf{(}SC(Ag_1, Ag_2, t, p \therefore p')$

$U^{+w}\ Attack\text{-}content(Ag_2, SC(Ag_1, Ag_2, t, p \therefore p'))\mathbf{))}$

In this section we defined the semantics of argumentation relations about propositional commitments. The argumention relation about the other types of commitments are related to the underlying propositional commitments. For example, the justification of a conditional commitment about a proposition is defined as the justification of the associated propositional commitment. Formally:

$$Justify\text{-}content(Ag_1, PCC(Ag_1, Ag_2, t, (p, p'))) =_{def}$$
$$Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, , p'))$$

7.4.8 Link between Commitments and Argumentation

Until now we gave the syntax and semantics of all the elements of our formalism. We can now formally establish the link between commitments and argumentation. This link is shown by the two following formulae.

Creation conditions

S66. $AG^+(Create(Ag_1, PC(Ag_1, Ag_2, t, p)) \Rightarrow$
$G^+\neg Contradict\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, p))$
$\wedge F^+(Challenge\text{-}content(Ag_2, PC(Ag_1, Ag_2, p))$
$\Rightarrow AX^+F^+ Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, p)))$
$\wedge F^+(Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, p))$
$\Rightarrow AX^+F^+ Defend\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, p))))$

This formula is a *rationality postulate* that we impose in the model. It provides the conditions generated by the creation of a commitment on all paths. The agent must be in a position to check these conditions before creating commitments. Indeed, if an agent creates a commitment, then it should not contradict itself during the conversation. It must also be able to justify its commitment if it is challenged and to defend it if it is attacked. By establishing the link between commitments and arguments, this formula reflects the deontic aspect of commitments. These conditions are also valid for withdrawal, acceptance and refusal because their semantics is expressed in terms of the creation operation.

Because this formula holds on all paths of the model, it seems to be strong. However, this formula is defined as a constraint that *software* conversational agents must respect. When an agent participates in a conversation using some protocol, it must respect this constraint. If not, we conclude that this agent does not respect the semantics. Therefore, it is easy to verify whether agents respect or not the semantics by verifying if they respect the different constraints. The protocol they use must also respect these constraints. In Chapter 8, we propose a model checking technique addressing this issue. Computationally speaking, agents' programs must include these constraints as rules, and the protocol can be implemented as a set of rules representing these constraints. In Chapter 9, we propose such an implementation using a set of dialogue games.

We notice that it is possible to relax this constraint by changing the model. The idea is to change the model when an agent creates a commitment (and in a general way when an agent performs an action). In this case, this constraint will hold on all paths of the new model and not of the original model. This means that, it is possible to capture, for example, the case in which an agent contradicts itself. However, our objective is not to model the different possibilities but to specify the constraints to be respected by agents. In other words, we are only interested in models respecting these constraints. In addition, changing the whole model increases the complexity of the model checking (see for example (Rao and Georgeff, 1993)).

On the other hand, an agent challenges a commitment content if it has no argument for or against this content. Therefore, an agent challenges a commitment content if it cannot accept or refuse such a commitment content. Formally:

Challenge conditions

S67. $AG^{+}((Active(PC(Ag_1, Ag_2, t, p))$
$\wedge \neg Accept\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, p))$
$\wedge \neg Refuse\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, p)))$
$\Rightarrow Challenge\text{-}content(Ag_2, PC(Ag_1, Ag_2, p)))$

7.5 Postulates

In this section we give some additional propositions (P) of our logical model. Proofs of these propositions are based on the semantics we defined in the previous section.

P1. $AG^{+}(Create(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$\neg X^{-}F^{-}(Active(SC(Ag_1, Ag_2, t, \varphi)))$
$\wedge (Active(SC(Ag_1, Ag_2, t, \varphi))$
U^{+}
$Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$

This formula states that if an agent creates a commitment then:

1. The commitment was never active in the past (thus it does not exist).
2. The commitment will hold until the moment of its withdrawal.

In other words, a commitment becomes active after its creation, and it remains active until its withdrawal.

Proof

If an agent creates a commitment which is already active, then according to S53 this commitment has already been created or reactivated. If the commitment is reactivated, then according to S51 and S37 it has already been created. However, this is not possible according to the semantics of the creation action (S36).

In addition, according to S53, a commitment is active iff it has already been created or reactivated and not yet withdrawn. Consequently, one can check if a commitment is active at a given moment on a path *Pa* by checking if it was already created in the past and if since its creation, it has been not withdrawn. Thus, the creation of a commitment implies that it is active until withdrawal.

□

P2. $AG^{+}(\neg Active(SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$\neg X^{+}Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)))$

Formula P2 indicates that if a commitment is not active, then it can not be withdrawn.

Proof
This formula is a consequence of formula S53 and the semantics of U^-. Let us suppose that the commitment is inactive at a given moment. Consequently, either this commitment was not created or reactivated in the past, or, since its creation or reactivation, the commitment was already withdrawn. In these two cases, the commitment cannot be withdrawn.
□

P3. $AG^+(Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$G^+(Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\wedge \neg Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\wedge \neg Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$

This formula states that if a commitment is satisfied, then it remains always satisfied and it cannot be violated or withdrawn.

Proof
According to the semantics of satisfaction (S40), *Satisfy* formula is satisfied in the model along a path *Pa* at any state of this path. Consequently, if it is satisfied, it remains always satisfied. Because the path *Pa* is a satisfaction path in the sens of the accessibility relation *Rsc*, the commitment cannot be violated along this path. In addition, according to S53, if an agent satisfies a commitment, then this commitment becomes inactive. Therefore, the commitment cannot be withdrawn
□

In the same way we can prove the following proposition:

P4. $AG^+(Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$G^+(Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\wedge \neg Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\wedge \neg Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi))))$

P5. $AG^+(Create(Ag_1, SC(Ag_1, Ag_2, t, \varphi) \vee Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))) \Rightarrow$
$X^+F^+ Violate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\vee X^+F^+ Satisfy(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$
$\vee X^+F^+ Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)))$

This formula indicates that if an agent creates or reactivates a commitment, then it must violate it, satisfy it, or still withdraw it. These operations can take place in the future of the moment following the creation of the commitment. The proof of this proposition follows from the semantics of theses operations.

P6. $AG^+(Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)) \Rightarrow$
$G^+ \neg SC(Ag_1, Ag_2, t, \varphi)$
$\vee$

$$(\neg SC(Ag_1, Ag_2, \varphi)\ U^+\ Reactivate(Ag_1, SC(Ag_1, Ag_2, \varphi)))$$

This proposition states that a commitment remains withdrawn until an *eventual* reactivation. Thus, the only authorized operation after the withdrawal of a commitment is its reactivation. The proof of this proposition follows from the semantics of Withdraw and Reactivate and from the meaning postulates M39.

We have also the following meaning postulates:

M6. $AG^+\neg(Active(SC(Ag_1, Ag_2, t, \varphi)) \wedge Active(SC(Ag_1, Ag_2, t', \neg\varphi)))$

This postulate states that it is not possible to have on a given path two active commitments of the same debtor whose contents are respectively φ and $\neg\varphi$.

M7. $AG^+ (Active(SC(Ag_1, Ag_2, t, \varphi) \,\&\, Accept\text{-}content(Ag_1, SC(Ag_2, Ag_1, t', \neg\varphi)) \Rightarrow$
$Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \varphi)))$

This formula indicates that if:
1. The agent Ag_1 already is committed that φ,
2. The commitment still holds,
3. This agent accepts the commitment of its interlocutor for $\neg\varphi$,

then this implies that the agent withdraws its commitment for φ. If Ag_1 does not withdraw this commitment, we would have two active commitments on a given path whose contents are φ and $\neg\varphi$. However, this is not possible according to M6.

7.6 Discussion

7.6.1 Meaning of Speech Acts

The meaning of some important speech acts, especially the ones commonly used in multi-agent interactions, can be expressed using our framework. According to illocutionary logic (Searle and Vanderveken, 1985), the five illocutionary points of language use are: the *assertive point*, *the commissive point*, *the directive point*, *the declaratory point* and *the expressive point*. The *assertive point* consists in representing how things are in the world. The *commissive point* consists in committing the speaker to doing something. The *directive point* consists in trying to get the hearer to do something. The *declaratory point* consists in doing something by way of representing oneself as doing it. The *expressive point* consists in expressing attitudes.

Assertive acts can be represented by propositional commitments and by conditional commitments about propositions. For example, the performance of an *Inform* act can be defined as the *creation* of a propositional commitment. The *inform* act $Inform(Ag_1, Ag_2, t, p)$ indicates that the speaker Ag_1 wants to inform the addressee Ag_2 that p is true. Formally, we can write:

$Inform(Ag_1, Ag_2, t, p) =_{def} Create(Ag_1, PC(Ag_1, Ag_2, t, p))$

The operations applied on the content of these commitments can be considered as assertive or directive acts. For example, the *Assert* act $Assert(Ag_1, Ag_2, t, p)$ means that the speaker Ag_1 is committed relatively to the addressee Ag_2 that p is true. In our framework, this acts can be defined by the acceptance of a commitment content in the context where this commitment exists. Formally:

$Assert(Ag_2, Ag_1, t, p) =_{def} Accept\text{-}content(Ag_2, PC(Ag_1, Ag_2, , p))$

The assertive act about an argument can be defined by a justification relation:

$Assert((Ag_2, Ag_1, t, p \therefore p')) =_{def} Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, , p), p')$

Commissive acts can be reflected by the action commitments and the conditional commitments about actions. The point of the commissive acts is to commit the debtor, relative to the creditor, to the performance of an action α with or without a certain condition. The performance of the action α makes a proposition p true. For example, a *promise* act $Promise(Ag_1, Ag_2, t, \alpha, p)$ means that agent Ag_1 is committed towards agent Ag_2 to do α without condition. This act can be defined either by the creation of an action commitment or by the acceptance of the content of a commitment attempt:

$$Promise(Ag_1, Ag_2, t, \alpha, p) =_{def} Create(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, p)) \lor Accept\text{-}content(Ag_1, ACT(Ag_2, Ag_1, , (\alpha, p)))$$

Directive acts can be represented by commitment attempts and by challenges of commitment contents. The operations applied to the content of commitment attempts can be considered as assertive, commissive or directive acts. *Request* is an example of a directive act that can be defined in our framework as follows:

$Request(Ag_1, Ag_2, t, \alpha, p) =_{def} Create(Ag_1, ACT(Ag_1, Ag_2, t, (\alpha, p)))$

The *request* act $Request(Ag_1, Ag_2, t, \alpha)$ indicates the fact that agent Ag_1 asks agent Ag_2 to do α. If Ag_2 accepts the request, then it promises Ag_1 to do α (see the previous definition of the promise act).

A declaratory act brings about a state of affairs that makes its content true (Colombetti, 2000). An example of declaration is “the auction is open” that is used to open an auction. In our framework, a delaratory act can be captured by the immediate satisfaction of a propositional commitent. Formally:

$$Declare(Ag_1, Ag_2, t, p) =_{def} Create(Ag_1, PC(Ag_1, Ag_2, t, p)) \land Satisfy(Ag_1, PC(Ag_1, Ag_2, t, p))$$

Expressive acts can also be captured using propositional social commitments.

In this section we showed that our formalism handles in a unified framework both pragmatic and semantic issues of agent conversation. In addition, the framework can capture many different types of illocutionary acts according to speech acts theory. Since the framework makes it possible to capture all these aspects, it can be used as a powerful means to specify, model and implement flexible and highly expressive protocols for agent communication.

7.6.2 A Model-Theoretic Semantics for Defeasible Argumentation

According to several researchers in defeasible argumentation, using a model-theoretic semantics is not adapted to defining the meaning of the central notions of defeasible argumentation like attack, rebuttal, defense, etc. The purpose of this section is to show that such a model theory can be successfully used to capture the semantics of these notions.

According to Pollock (1991), Vreeswijk (1997), and Prakken and Vreeswijk (2000), the meaning of defeasible notions should not be found in a correspondence with reality by using a model theory, but in their role in dialectical inquiry. The reason is that these notions are not 'propositional', and consequently, their meaning is not naturally captured in terms of correspondence between a proposition and the world. We agree with the fact that the defeasible notions are not propositional, because in our framework they are actions applied to social commitments. Thus, these defeasible concepts (considered in this chapter as argumentation relations), can be captured in a model theory by using a dynamic logic within a global framework of temporal logic. Using these two logics enables us to represent the relation between arguments by taking into account the temporal and the dynamic characteristics of the argumentative interactions between agents. Our theoretical model semantics does not establish a correspondence between defeasible notions (as propositions) and the world, but defines the meaning of operations that agents can apply on their social commitments and the meaning of argumentative supports of these operations. This semantics allows us to capture the conditions on handling commitments and arguments (see S66 and S67). The branching temporal nature of our logical model makes it possible to capture the fact that an agent in a given state at a given moment has several strategies. Agents use their argumentation systems to choose a strategy among others.

On the other hand, the nonmonotonicity property of arguments can be captured in a model theory of branching temporal logic. The idea is that an argument is valid only in a given state, at a given moment for a given agent. An argument is not valid (not satisfied in the Kripke model) when it is attacked and cannot be defended. This idea can be formulated as a property in our logical model by using the path quantifiers A and E (see M65 (1), S65 (2)). In addition, an advantage of using a model theory of temporal and dynamic logics to define the semantics is that we can then use model-checking techniques (Clarke et al., 1986). These techniques enable us to verify some interesting properties of the formalism. In this context, we can use our approach to specify interaction protocols illustrating how agents interact by acting on commitments and on arguments. The automatic tools of model checking (called model checkers) make it possible to provide simulations and traces of execution of such protocols in order to verify properties that these protocols must satisfy (Clarke et al., 2000), (Wooldridge et al., 2002). These techniques are not offered for a logic based on dialectical systems.

In the context of agent interactions, using only an argumentative-based semantics is not sufficient to capture the nonmonotonic reasoning of agents. The reason is that in their conversations, agents do not use only an acceptance theory based on arguments and on attack and defense relations. Agents must also take into account social relations such as trustworthiness.

Finally, we think that a model theoretical semantics and a dialectical-based semantics are not contradictory but rather complementary in the context of agent communication. A model theoretical semantics using temporal and dynamic logics has the advantage of capturing actions and temporal issues of communicative acts. Dialectical-based semantics have the advantage of representing the interaction between arguments that agents use in their conversations.

7.6.3 Related Work

Semantical considerations for agent interaction have recently begun to find a significant audience in the MAS community. We can distinguish four kinds of semantics for agent interactions:

1- Mentalistic semantics: This subjective semantics is based on so-called agent's mental states (e.g. beliefs, desires and intentions). The best-known formalisms describing it are: Cohen and Levesque's intention logic (1990), Rao and Georgeff's BDI framework (1995), and the KARO framework proposed by van Linder et al. (1998). KQML (Finin et al., 1995) and FIPA-ACL (FIPA, 1997, 1999, 2001a) use this type of semantics to define a pre/post conditions semantic of communication acts. For example, the semantics of a KQML message is given by the following three ingredients: 1) a precondition on the mental states of the sender and the receiver before the communication of the message, 2) a postcondition that should hold after the communication and 3) a completion condition that indicates when the perlocutionary effect has been fulfilled. The advantage of this semantics is its compatibility with the formalisms used for reasoning about rational agents. Hence, the same formalism can be used to specify the agents' mental states and the communication acts they perform. However, the verification of such a semantics is not possible if we cannot access to the agents' programs. In this situation we cannot verify whether the agents' behavior matches their private mental states. In this context, van Eijk and his colleagues (2003) proposed a verification method for agent communication using a framework called Agent Communication Programming Language (ACPL) (van Eijk et al., 2001). ACPL is designed to program systems of agents that communicate by exchanging information. The authors consider the operational semantics of this language which describes the agents' behavior in terms of their computations. From this semantics, they identified a notion of observable behavior that captures those aspects of computations that are visible to an external observer, and they introduced an assertion language to express specifications of this behavior. To check if agents act in accordance with the behavior specification, the authors developed a verification calculus based on a compositional proof system.

Another limitation of KQML is the pre/post condition semantics. This semantics offers no dynamic or operational description of agent interactions. Because our approach is based on public and argumentative concepts, the compliance verification can be made without having access to the agents' programs. The satisfaction and the violation of agents' commitments make it possible to determine if the agent respects our semantics. In addition, the agents' ability to argue and to justify their commitments facilitates this verification. Moreover, our semantics treats more explicitly the dynamic aspect of agent communication. This aspect is modeled not only by the agents' actions on commitments and on their contents and by the argumentation relations, but also by the evolution of commitment states and commitment content states.

2- Social semantics: This objective semantics was proposed by Singh as an alternative to the mentalistic one (Singh, 2000). It is based on social commitments and it stresses the importance of conventions and the public aspects of agent interactions. Singh used CTL to propose a formal language and a formal model in which the notion of commitment is described by using an accessibility relation. Verdicchio and Colombetti proposed a logical model of social commitments by extending CTL* (Verdicchio and Colombetti, 2003). They introduced a number of predicates in order to represent events and actions. They specified some axioms to model agents that create commitments, create precommitments, and accept precommitments. They also studied the fulfillment and violation of commitments. Mallya et al. (2004) used the temporal commitment structure specified by Fornara and Colombetti (2002) to define some constraints in order to capture some operations on commitments. They dealt with temporal commitments by studying their satisfactions and breaches. Our logical model belongs to this class of semantics, but it differs from these proposals in the following respects:

a) In our approach the commitment semantics is defined as an accessibility relation that takes into account the satisfaction of the commitment. The commitment semantics is defined in terms of the paths along which the commitment must be satisfied. This way is more intuitive than the semantics defined by Singh.

b) We differentiate commitments as static structures evaluated in states from the operations applied to commitments as dynamic structures evaluated on paths. This enables us to describe more naturally the evolution of the communication as a system of states / transitions which reflects the interaction dynamics. Thus, our logical model allows us to describe the dynamics of agent interactions in terms of the actions that agents apply to commitments, commitment contents and to arguments. These actions are captured by the *perform* operator used in dynamic logic and that we introduce in our model.

c) In our model, the strength of commitments as a basic principle of agent communication does not result only from the fact that they are observable, but also from the fact that they are supported by arguments. The social commitment notion we formalize is not only a public notion but also a deontic one. The deontic aspect is captured by the fact that commitments are thought of as obligations. The agent is obliged to respect its commitments (i.e to satisfy them), to behave in accordance with these commitments and to justify them. The idea is to impose this constraint in the model we are interested in. The agent is also obliged not to contradict its commitment contents during the conversation. The creation operation and the argumentation relations capture this deontic aspect. Formulae S66 and S67 which supplement our semantics show how our approach makes it possible to capture this aspect. Indeed, the link we establish between commitments and arguments enables us to formally express the following idea: by committing towards other agents that a certain formula is true, the agent is compelled not to contradict itself during the conversation. It must also be able to explain, argue, justify and defend itself if another participant contradicts it.

d) In our semantics, we capture not only propositional commitments, but the various othertypes of commitments. This enables us to have a greater expressivity and to capture many different types of speech acts. In addition, all the elements constituting our commitment and argument approach are expressed using the same logical framework. The different types of commitments, the different operations on them, and the different argumentation relations are semantically specified in a clear and unambiguous way.

3- Argumentation-based semantics: This type of semantics is defined in (Amgoud et al., 2002), (Parsons et al., 2002), (Parsons et al., 2003) to capture the meaning of certain communication acts. It is based upon an argumentation system in which the agents' reasoning capabilities are often linked to their ability to argue. These reasoning capabilities are mainly based on the agent's ability to establish a link between different facts, to determine if a fact is acceptable, to decide which arguments support which facts, etc. The authors proposed a two-layered semantics. The first layer captures the reasoning level of agents. Agents must check some preconditions in order to use a communication act. These preconditions are described in terms of arguments. For example, before using an assertion act that *p*, an agent checks whether it has an argument in favor of *p*. The second layer relies upon the formal dialectics introduced by Mackenzie (1970). Dialectical models are rule-governed structures of organized conversations in which two parties (in the simplest case) speak in turn in an orderly way. These models associate to each agent a commitment store (CS), which holds the information given by the interlocutors during the dialogue. This layer describes the rules which define how the CS is updated. For example, after an assertion act that *p* is true, the CS of the speaker is updated by adding p to it. This semantics has the advantages of being simple and of taking into account the argumentation aspect of agent communication. In addition to the fact that this semantics does not take into account the temporal and dynamic aspects of communicative acts in its formalization, it is different from our approach on several points. The fact that it uses a logic without theoretical model makes a formal verification impossible. On the other hand, the semantics is described in terms of pre/post conditions and it does not capture the meaning of the different communication acts. The commitment notion used in this semantics is different from the one we use in our semantics. In Amgoud et al's approach, this concept captures only the propositions stated by the agents. Contrary to our approach, the satisfaction, violation, cancellation and reactivation notions do not appear. Moreover, in terms of argumentation, only the argue operation is captured. The attack and defense operations are not addressed in this semantics. Finally, the dynamic aspect of agent communication is reduced to the sole update operations of the CS. These operations reflect only the history without clearly reflecting the current state of the communication. On the other hand, in our approach this state is well captured by the states of different commitments and arguments handled in the conversation.

4- Protocol based semantics: Developed by Pitt and Mamdani (2000), this type of semantics is based on the notion of protocol. The communication between two (or more) agents is viewed as a conversation. The meaning of communication acts is specified by describing an input-output relationship. The meaning of a speech act (as input) is defined to be the intention to perform another speech act (as output). This meaning then matches the set of the possible following answers. This semantics has the advantage of taking into account the context and the conversation state. However, technically, protocols are used as a practical tool and not as a means to define semantics. For example, by using only protocols, we cannot define the meaning of some notions like satisfaction, violation, contradiction, justification, etc. Protocols must be specified in accordance with a given semantics in such a way that a compliance verification is possible. In our approach we can define protocols by using our semantics and verify whether some properties (that we have to specify yet) are satisfied. For instance, such a property can be stated as follows: "It is not possible to withdraw a commitment that is previously satisfied". Because our semantics is expressed in a temporal logic, we can use protocols specifying that an action cannot take place before another. For example, a commitment cannot be cancelled before its creation. A protocol can also specify that when a commitment is created by an agent Ag_1, several paths are possible for its interlocutor Ag_2 (acceptance, refusal, challenge). However, the choice of the path cannot be made without returning to the semantics. For example, acceptance indicates that the agent is also committed towards the accepted content. Thus, agent Ag_2 must be able to justify this content and to satisfy the commitment.

Finally, we notice that although our accessibility relation *Rsc* is a dynamic function, we do not need to change the Kripke model *M* to capture this dynamics. This way of modeling is different from that used for example in KARO framework (Meyer et al., 1999). In KARO, the whole Kripke model must be changed as illustrated by the following formula:

$$M, s \models \langle do_i(\alpha)\rangle\varphi \text{ iff } \exists M', s' \,(M', s' \in r(i, \alpha)(M, s) \,\&\, M', s' \models \varphi)$$

Where $\langle do_i(\alpha)\rangle\varphi$ represents the fact that agent *i* has the opportunity to do the action α and that doing α leads to φ, and *r is a function defined from another function* r_0 *as follows*:

$$r_0: A \times At \rightarrow (S \cup \{\varnothing\}) \rightarrow (S \cup \{\varnothing\})$$

where *A* is a set of agents, *At* is a set of atomic actions and *S* a set of states. $r_0(i, \alpha)(s)$ yields the (possibly empty) state transition in *s* caused by the event $do_i(\alpha)$. A successful performance of an atomic action always results in a state transition to another state in the model. *r* is defined as follows:

$$r(i, \alpha)(M, s) = M, r_0(i, \alpha)(s)$$

$r(i, \alpha)(M, s)$ yields the model change and the state transition caused by the event $do_i(\alpha)$. In our model that fits in naturally with the use of CTL* the whole dynamics is represented in one unique model. Thus, we do not need to define a function like r_0. Indeed, we can capture all the actions that agents apply to commitments and to their contents without changing the model, but simply by changing the states of the model. This solution increases the number of states in the model. However, it enables us to reduce the complexity of the underlying decision procedure, and it gives rise to more efficient model-checking.

7.7 Conclusion

In this chapter, we developed a logic and formal semantics for our pragmatic approach based on commitments and arguments to model agents' interactions. We proposed a logical model based on a combination of CTL* and dynamic logic. The model captures the different commitment types, the different actions that agents apply to these commitments and the various argumentation relations. In addition, the model captures the link between commitments and arguments that enables us to express the deontic aspect of commitments. Our semantic framework can also be used to express the meaning of some important speech acts, especially the ones commonly used in multi-agent interactions. Finally, we argued that our model-theoretic semantics can be successfully used to capture the semantics of defeasible arguments.

Chapter 8*

A Tableau Method for Verifying Dialogue Game Protocols (a Model Checking Approach)

In this chapter, we address the problem of verifying dialogue game protocols using a tableau-based model checking technique. These protocols are specified using the $DCTL^*_{CAN}$ logic that we developed in Chapter 7. Unlike the model checking algorithms proposed in the literature, the algorithm that we propose in this chapter allows us not only to verify if the dialogue game protocol (the model) satisfies a given property expressed in $DCTL^*_{CAN}$, but also if this protocol respects the tableau rules of the communicative acts. This algorithm is an on-the-fly efficient algorithm.

8.1 Introduction

As outlined in Chapter 3, dialogue games provide an interesting way of specifying agent communication protocols (see for example (Dastani et al., 2000), (McBurney and Parsons, 2002), (Maudet and Chaib-draa, 2002), (Bentahar et al., 2004a, 2004d)). These games aim at offering more flexibility by combining different small games to construct complete and more complex protocols. Dialogue games can be thought of as interaction games in which each agent plays a move in turn by performing utterances according to a pre-defined set of rules.

From another point of view, formal verification methods became usable by industry quite recently and there is a growing demand for professionals able to apply them (Huth and Ryan, 2000). We can think of formal verification techniques as being composed of three parts:

- A framework for modeling systems, typically a description language.
- A specification language for describing the properties to be verified.
- A verification method to establish whether the description of a system satisfies the specification or not.

* We would like to thank John-Jules Ch. Meyer from Utrecht University, Intelligent Systems Group, Girish Bhat from Cosine Communication Inc., Rance Cleaveland from State University of New York at Stony Brook, and Yves Lespérance from York University for their interesting comments and extremely helpful discussions about the matter presented in this chapter.

The two main approaches to verify system properties are proof-based approaches and model-based approaches. In the proof-based approaches, the system description is a set of logical formulae Γ and the specification is another formula ϕ. The verification method consists of trying to find a proof that $\Gamma \vdash \phi$. This typically requires guidance and expertise from the user in order to identify suitable lemmas and auxiliary assertions. In the model-based approaches, the system is represented by a finite model M using an appropriate logic. The specification is again represented by a formula ϕ and the verification method consists of computing whether the model M satisfies ϕ or not. This is usually done automatically.

Model-based techniques rely on models describing the system's possible behaviors in a mathematical precise and unambiguous manner (Queille and Sifakis, 1981), (Lichtenstein and Pneuli, 1985), (Clarke et al., 1986). The system models are accompanied by algorithms that systematically explore all the states of the system model. This provides the basis for a whole range of verification techniques ranging from an exhaustive exploration (model checking) to experiments with a restrictive set of scenarios in the model (simulation), or in reality (testing).

Recently, the verification of MAS has become an attractive field of research. Several proposals have been put forward for model checking MAS. Some of these proposals use existing model checkers (for example SPIN and JPF2) by translating some agent specification languages (for example MABLE and AgentSpeak) to the languages used by these model checkers (Wooldridge et al., 2002), (Bordini et al., 2003a, 2003b). Other proposals adapt some model checking techniques (for example bounded and unbounded model checking) and propose new algorithms for verifying temporal and epistemic properties of MAS (Penczek and Lomuscio, 2003), (Kacprzak and Penczek, 2004a, 2004b), (Raimondi and Lomuscio, 2004).

In the domain of agent communication, only some research work tried to address the verification of agent communication protocols. Endriss and his colleagues (2003) dealt with the problem of checking and possibly enforcing conformance to agent communication protocols. Huget and Wooldridge (2004) addressed the problem of checking that agents correctly implement the semantics of an agent communication language. Walton (2004) applied model checking techniques in order to verify the correctness of a communication protocol. Baldoni and his colleagues (2004) tackle some aspects of the conformance verification, i.e. the verification that a given protocol implementation conforms to its abstract specification. Giordano and her colleagues (2004) addressed the problem of specifying and verifying systems of communicating agents in a Dynamic Linear Time Temporal Logic (DLTL).

Except the work done by (Giordano et al., 2004), all the other work on model checking of MAS are based only on temporal and epistemic logics. In this chapter, we propose a model checking-based verification of dialogue game protocols using a temporal and dynamic logic. These protocols are specified as transition systems using our $DCTL^*_{CAN}$ logic (Dynamic and CTL* logic for Commitment and Argument Network) that we developed in Chapter 7. In contrast to (Giordano et al., 2004), the dynamic aspect of our logic is represented by action formulae and not by strengthening the *until* operator by indexing it with the regular programs of dynamic logic. Our protocols are specified as actions that agents apply to social commitments (SC) and to SC contents. In addition, the model checking procedure that we propose allows us to verify not only that the dialogue game protocol (the theoretical model) satisfies a given property, but also that the tableau semantics of the communicative acts is respected. The idea is to integrate this semantics in the specification of the protocol, and then to propose a parsing method to verify that the protocol specification respects the semantic definition. Consequently, if agents respect these protocols, then they also respect the semantics of the communicative acts. We have here a mechanism for checking the agents' compliance with the semantics without taking into account the agents' specifications created by the developers. Indeed, we have only one procedure to verify: 1) the correctness of the protocols relative to the properties that the protocols should satisfy; and 2) the conformance of agents to the semantics of the communicative acts. However, the tableau semantics (the tableau rules) we use in this chapter describe only the structure of the commitment formulae and not the semantics as defined in Chapter 7. The purpose of this technique is to verify the temporal properties of the protocol and to ensure that the structures of the commitments are the same in both the protocol and the specification. The advantage of verifying the structures of the commitments is to ensure that all agents participating in a communication share the same description of the communicative acts. In addition this technique based on the tableau method can be generalized to also verify the semantic definitions proposed in Chapter 7. This work goes beyond the objectives of this thesis and will be a priority subject of our future research.

To our knowledge, until now there is no work that addressed the verification problem of dialogue game protocols. Indeed, the contributions of this chapter are:

1- A formulation of dialogue game protocols using transition systems. This formulation enables us to represent not only the allowed communicative acts but also the underlying tableau semantics.
2- An automata and tableau-based technique to check if a protocol satisfies the specifications and the structure. These two verifications are done at the same time.

The rest of this chapter is organized as follows. Section 8.2 introduces the model checking problem and a class of algorithms based on the tableau method to which our procedure belongs. Section 8.3 presents a tableau semantics of our $DCTL^*_{CAN}$ logic. Section 8.4 defines the transition systems that we use to model dialogue game protocols and the underlying semantics. The problem of verifying these protocols is addressed in Section 8.5. In this section, we present the Alternating Büchi Tableau Automata, the translation procedure of temporal and action formulae to this automata, and the model checking algorithm. Proofs of different properties are also presented in this section. Section 8.6 presents related work and Section 8.7 concludes the chapter.

8.2 Model-Checking Overview

8.2.1 Automata-Theoretic Approach

The model-checking problem for a branching temporal logic is as follows: Given a *Kripke structure* K and a branching temporal formula ψ, determine if $K \models \psi$. The state space of a transition system can be thought of as a Kripke structure. For linear temporal logics, a close and fruitful connection with the theory of automata on infinite words has been developed (Vardi and Wolper, 1986), (Courcoubetis et al., 1992). The basic idea is to associate with each linear temporal logic formula a finite automaton on infinite words that accepts exactly all the computations that satisfy the formula. For these logics, each Kripke structure may correspond to infinitely many computations. Model checking is thus reduced to check inclusion between the set of computations allowed by the Kripke structure and the language of an automaton describing the formula (Vardi and Wolper, 1986). For branching temporal logics, each Kripke structure corresponds to a single non-deterministic computation. On that account, model checking is reduced to check the membership of this computation to the language of the automaton describing the formula (Wolper, 1989). For these logics, the automata-theoretic counterpart is automata on infinite trees. By reducing the satisfiability to the non-emptiness problem for these automata[6], optimal decision procedures have been obtained for various branching temporal logics (Emerson and Lei, 1986), (Vardi and Wolper, 1986), (Emerson and Sistla, 1984), (Courcoubetis et al., 1992).

Bernholtz, Vardi, and Wolper (1994) argued that *alternating tree automata* are the key to a comprehensive and satisfactory automata-theoretic framework for branching temporal logics. Alternating tree automata on infinite trees generalize the standard notion of non-deterministic tree automata by allowing several successor states to go down along the same branch of the tree (Muller and Schupp, 1987). Tree automata generalize sequential automata in the following way: on a given binary tree, the automaton starts its computation at the root in an initial state and then simultaneously works down the paths of the tree level by level. The transition relation specifies the two states that are the two sons of a node. The tree automaton accepts the tree if there is a run built up in this fashion which is *successful*. A run is successful if all its paths are successful in a sense given by an acceptance condition for sequential automata.

It is known that while the translation from branching temporal logic formulae to non-deterministic tree automata is exponential, the translation to alternating tree automata is linear (Muller et al., 1988). This explains the efficiency of model checking for these logics. Thus, alternating tree automata provide a unifying and optimal framework for both satisfiability and model-checking problems for branching temporal logics.

---

[6] The non-emptiness problem for automata is to decide, given an automaton A, whether its language $L(A)$ is non-empty. The language $L(A)$ is the set of words accepted by A.

The model checking approach that we use for our logic is based on an alternative view of model checking proposed by Bhat and Cleaveland (1996, 2001). This view relies on translating formulae into intermediate structures, Alternating Büchi Tableau Automata (ABTA). Unlike the other model checking techniques, this technique allows us to verify not only temporal formulae, but also action formulae. Because our logic is based on an action theory, this technique is more suitable. This approach is a tableau-based model checking. The following section introduces this approach.

8.2.2 Tableau-based Algorithms for Model-Checking

Tableau-based algorithms are based on the use of assertions and *proof rules*. Assertions are typically of the form $s \vdash_M \phi$ and mean that state *s* in model *M* satisfies the formula ϕ. Using a set of proof rules we aim to prove the truth or falsity of assertions. But unlike traditional proof systems which are bottom-up approaches, tableau-based algorithms work in a *top-down* or *goal-oriented* fashion. Proof rules are used in order to prove a certain formula by inferring when a state in a Kripke structure satisfies such a formula. According to this approach, we start from a goal, and we apply a proof rule and determine the sub-goals to be proven. The proof rules are designed so that the goal is true if all the sub-goals are true. The advantage of this method is that the state space is explored in a need-driven fashion. The algorithm searches only the part of the state space that needs to be explored to prove or disprove a certain formula. The state space is constructed while the algorithm runs. This kind of algorithms, also referred to *on-the-fly* or *local* algorithms, have been found to be useful in practice since in many cases only a small part of the state space needs to be explored to prove a formula (Cleaveland, 1990), (Stirling and Walker, 1991), (Bhat and Cleaveland, 2001).

The tableau-based algorithms proposed in (Cleaveland, 1990), (Stirling and Walker, 1991) have exponential time and space complexity. The exponential penalty incurred by these algorithms is mainly due to the fact that these algorithms work by constructing proof trees. Like (Bhat, 1998), the algorithm that we use for our $DCTL^*_{CAN}$ logic avoids this exponential penalty by using graphs instead of trees to represent proofs.

The tableau decision algorithm that we use provides a systematic search for a model which satisfies a particular formula of our logic. It is a graph construction algorithm. Nodes of the graph are sets of $DCTL^*_{CAN}$ formulae and *tableau rule* names. Tableau rules are inference rules designed so that the formula is true if all the sub-formulae are true. The main difference between proof rules and tableau rules is that proof rules work on assertions, while tableau rules work on logical formulae. The difference between assertions and logical formulae is that logical formulae are written without taking into account the states of the model. The interpretation of vertex labeling is that for the vertex to be satisfied, it must be possible to satisfy all the formulae in the set together. Each edge in the graph represents a satisfaction step of the formula contained in the starting vertex. These steps correspond to the application of a set of tableau rules. These rules express how the satisfaction of a particular formula (the goal) can be obtained by the satisfaction of its constituent formulae (sub-goals).

8.3 Tableau rules for DCTL*$_{CAN}$

The semantics we use here is a tableau semantics (Cleaveland, 1990) that we can consider as a simplification of the semantics that we defined in Chapter 7. This semantics is specified in terms of the decomposition of formulae to sub-formulae using a set of tableau rules. These rules are given in Figures 8.1, 8.2, 8.3 and 8.4. For simplification reasons, we use in this chapter a simplified version of DCTL*$_{CAN}$ that is sufficient for the specification of dialogue game protocols. For example we consider only propositional and action commitments, and we do not consider formulae of commitment states and formulae of contradiction of commitment contents. In addition, to simplify the *Challenge-content* formula, we introduce a syntactical operator ?. Syntactically, $?\psi$ means that, a given agent does not know whether ψ is true or not.

$$R1\ \wedge : \frac{\psi_1 \wedge \psi_2}{\psi_1 \quad \psi_2} \qquad R2\ \vee : \frac{\psi_1 \vee \psi_2}{\psi_1 \quad \psi_2} \qquad R3\ \vee : \frac{E(\psi)}{\psi} \qquad R4\ \neg : \frac{\neg\psi}{\psi} \qquad R5\ ? : \frac{?\psi}{\psi}$$

$$R6\ \neg : \frac{A(\Phi)}{E(\neg\Phi)}$$

Figure 8.1. Tableau rules for propositional and universal formulas

The tableau semantics enables us to define top-down proof systems. The idea is: given a formula, we apply a tableau rule and determine the sub-formulae to be proven. Tableau rules are inference rules used in order to prove a formula by proving all the sub-formulae. The labels of these rules are the labels of states in the automata constructed from a given formula. For example, rule *R*1 of Figure 8.1 labeled by "$\wedge$" indicates that ψ_1 and ψ_2 are the two sub-formulae of $\psi_1 \wedge \psi_2$. This means that, in order to prove that a state labeled by "$\wedge$" satisfies the formula $\psi_1 \wedge \psi_2$, we have to prove that the two children of this state satisfy ψ_1 and ψ_2 respectively. This idea will be detailed in Section 8.5.1 when we will define the alternating Büchi tableau automata. According to rule *R*2, in order to prove that a state labeled by "$\vee$" satisfies the formula $\psi_1 \vee \psi_2$, we have to prove that one of the two children of this state satisfies ψ_1 or ψ_2. Rule *R*3 labeled by "$\vee$" indicates that ψ is the sub-formula to be proved in order to prove that a state satisfies E(ψ). According to rule *R*4 (resp. *R*5), the formula $\neg\psi$ (resp. $?\psi$) is satisfied in a state labeled by "$\neg$" (resp. ?), if this state has a successor representing ψ. Rule *R*6 is defined in the usual way where Φ is a set of path formulae.

The label "<α_ϕ>" (rule *R*7 of Figure 8.2) is the label associated with the action α whose performance makes the proposition ϕ true (see Chapter 7). According to this rule, in order to prove that a state labeled by "<α_ϕ>" satisfies *Perform*(α)ϕ, we have to prove that an accessible state via a transition labeled by *Perform*(α) satisfies ϕ. Rule *R*8 is defined using the same idea. The label "<*C*>" (rule *R*9) is the label associated with the creation action of a social commitment SC. According to this rule, in order to prove that a state satisfies *Create*(Ag_1, *SC*(Ag_1, Ag_2, *t*, ϕ)), we have to prove that an accessible state via a transition labeled by the creation action satisfies the sub-formula $SC(Ag_1, Ag_2, t, \phi)$. The rules R10 to R21 are defined in the same way.

$$R7 < \alpha_{\phi} >: \frac{E(\Phi, Perform(\alpha)\phi)}{E(\Phi, \phi)} \qquad R8 < \alpha_{SC} >: \frac{E(\Phi, Perform(\alpha)SC(Ag_1, Ag_2, t, \phi))}{E(\Phi, SC(Ag_1, Ag_2, t, \phi))}$$

$$R9 < C >: \frac{E(\Phi, Create(Ag_1, SC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, SC(Ag_1, Ag_2, t, \phi))}$$

$$R10 < W >: \frac{E(\Phi, Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, \neg SC(Ag_1, Ag_2, t, \phi))}$$

$$R11 < S_{PC}^{Ag1} >: \frac{E(\Phi, Satisfy(Ag_1, PC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, \phi)}$$

$$R12 < S_{AC}^{Ag1} >: \frac{E(\Phi, Satisfy(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \phi)))}{E(\Phi, Perform(\alpha)\phi)}$$

$$R13 < V_{PC}^{Ag1} >: \frac{E(\Phi, Violate(Ag_1, PC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, \neg\phi)}$$

$$R14 < V_{AC}^{Ag1} >: \frac{E(\Phi, Violate(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \phi))))}{E(\Phi, \neg Perform(\alpha)\phi)}$$

$$R15 < Rea >: \frac{E(\Phi, Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, SC(Ag_1, Ag_2, t, \phi))}$$

$$R16 < Ch >: \frac{E(\Phi, Challenge-content(Ag_2, PC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, PC(Ag_2, Ag_1, t', ?\phi))}$$

$$R17 < Acc >: \frac{E(\Phi, Accept-content(Ag_2, SC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, SC(Ag_2, Ag_1, t', \phi))}$$

$$R18 < Ref >: \frac{E(\Phi, Refuse-content(Ag_2, SC(Ag_1, Ag_2, t, \phi)))}{E(\Phi, SC(Ag_2, Ag_1, t', \neg\phi))}$$

$$R19 < Jus >: \frac{E(\Phi, Justify-content(Ag_1, PC(Ag_1, Ag_2, t, \phi), \phi'))}{E(\Phi, PC(Ag_1, Ag_2, t', \phi' \therefore \phi))}$$

$$R20 < Att >: \frac{E(\Phi, Attack-content(Ag_2, PC(Ag_1, Ag_2, t, \phi), \phi'))}{E(\Phi, PC(Ag_2, Ag_1, t', \phi' \therefore \neg\phi))}$$

$$R21 < Def >: \frac{E(\Phi, Defend-content(Ag_1, PC(Ag_1, Ag_2, t, \phi), \phi'))}{E(\Phi, PC(Ag_1, Ag_2, t', \phi' \therefore \phi))}$$

Figure 8.2. Tableau rules for action formulas

$$R22\ [PC_{Ag1}]: \frac{E(\Phi, PC(Ag_1, Ag_2, t, \phi))}{E(\Phi, \phi)}$$

$$R23\ [AC_{Ag1}]: \frac{E(\Phi, AC(Ag_1, Ag_2, t, (\alpha, \phi)))}{E(\Phi, Perform(\alpha)\phi)}$$

Figure 8.3. Tableau rules for commitment formulas

$$R24 \;<\equiv>: \frac{E(\Phi, l)}{l,\ E(\Phi)} \qquad R25 \;\wedge: \frac{E(\Phi, \varphi_1 \wedge \varphi_2)}{E(\Phi, \varphi_1, \varphi_2)}$$

$$R26 \;\vee: \frac{E(\Phi, \varphi_1 \vee \varphi_2)}{E(\Phi, \varphi_1)\ \ E(\Phi, \varphi_2)} \qquad R27 \;?: \frac{E(\Phi, ?\psi)}{E(\Phi, \psi)}$$

$$R28 \;X^- : \frac{E(\Phi, X^- \varphi_1, \ldots, X^- \varphi_n)}{E(\Phi, \varphi_1, \ldots, \varphi_n)} \qquad R29 \;X^+ : \frac{E(\Phi, X^+ \varphi_1, \ldots, X^+ \varphi_n)}{E(\Phi, \varphi_1, \ldots, \varphi_n)}$$

$$R30 \;\wedge : \frac{E(\Phi, \varphi_1 \therefore \varphi_2)}{E(\Phi, \varphi_1, X^+(\neg \varphi_1 \vee \varphi_2))}$$

$$R31 \;\vee: \frac{E(\Phi, \varphi_1 U^- \varphi_2)}{E(\Phi, \varphi_2)\ \ E(\Phi, \varphi_1, X^-(\varphi_1 U^- \varphi_2))} \qquad R32 \;\vee: \frac{E(\Phi, \varphi_1 U^+ \varphi_2)}{E(\Phi, \varphi_2)\ \ E(\Phi, \varphi_1, X^+(\varphi_1 U^+ \varphi_2))}$$

Figure 8.4. Tableau rules for state formulas

Rule *R*22 of Figure 8.3 indicates that E(ϕ) is the sub-formula of the formula E(*PC*(Ag_1, Ag_2, *t*, ϕ)). Thus, in order to prove that a state satisfies E(*PC*(Ag_1, Ag_2, *t*, ϕ)), we have to prove that the accessible state via a transition labeled by "[PC_{Ag1}]" satisfies E(ϕ). In the same way, we define the rule *R*23.

Finally, the rules *R*24 to *R*32 of Figure 8.4 are defined in the usual way. For example, according to rule *R*29, in order to prove that a state satisfies E($X^+\varphi$), we have to prove that the next state via the transition labeled by "X^+" satisfies the sub-formula E(φ).

8.4 Dialogue Game Protocols as Transition Systems

8.4.1 Specification

In this section we define the theoretical model of our model checking procedure. This model specifies the dialogue game protocols. These protocols are specified as a set of rules describing the entry condition, the dynamics and the exit condition of the protocol (Bentahar et al., 2004a) (this aspect will be detailed in Chapter 9). These rules can be specified in our logic as action formulae (actions on SC, actions on SC contents and argumentation relations). We define these protocols as transition systems. The purpose of these transition systems is to describe not only the sequence of the allowed actions (classical transition systems), but also the semantics of these actions and the semantics of the different elements used in our commitment and argument-based approach. The states of these transition systems are sub-transition systems (called semantic transition systems) describing the semantics of the actions labeling the entry transitions. Defining transition systems in such a way allows us to verify:
1- The correctness of the protocol (if the model of the protocol satisfies the properties that the protocol should specify).
2- The compliance to the semantics of the communicative actions (if the specification of the protocol respects the semantics).

In this chapter, we propose a model checking procedure in order to verify both (1) and (2) at the same time.

The definition of the transition system of dialogue game protocols is given by the following definitions:

Definition 8.1 *A semantic transition system T' describing the semantics of an action formula is a 6-tuple* $<S', Lab', F, Ls', R, \rightarrow, s'_0>$ *where*:
S' is a set of states,
$Lab' : S' \rightarrow 2^{\Phi p}$ *is the labeling state function, where* Φp *is the set of atomic propositions,*
F is a sub-set of the set of formulae from $DCTL^*_{CAN}$ (*F does not include the action formulae i.e. Create, Satisfy, Accept-content, etc.*),
$Ls' : S' \rightarrow F$ *is a function associating to each state a formula from* $DCTL^*_{CAN}$,
$R \in \{\wedge, \vee, \neg, ?, <\equiv>, X^+, X^-, PC_{Ag}, AC_{Ag}\}$ *is the set of tableau rule labels* (*without the rules for action formulae*),
$\rightarrow \subseteq S' \times R \times S'$ *is the transition relation,*
s'_0 *is the start state.*

Intuitively, states *s'* contain the sub-formulae of the action formulae, and the transitions are labeled by operators associated with the formula of the starting state. Semantic transition systems enable us to describe the semantics of formulae by sub-formulae connected by logical operators. Thus, there is a transition between states s'_i and s'_j iff $L'(s'_j)$ is a sub-formula or an semantically equivalent formula of $L'(s'_i)$. Following traditional usage we write $s \rightarrow^r s'$ instead of $<s, r, s'> \in \rightarrow$ where $s, s' \in S'$ and $r \in R$.

Definition 8.2 *A transition system T for a dialogue game protocol is a 6-tuple* $<S, Lab, \wp, L, Act, \rightarrow, s_0>$ *where*:
S is a set of states,
$Lab : S \rightarrow 2^{\Phi p}$ *is the labeling state function, where* Φp *is the set of atomic propositions,*
$\wp$ *is a set of semantic transition systems with* $\varepsilon \in \wp$ *is the empty semantic transition system,*
$L : S \rightarrow \wp$ *is the function associating to a state* $s \in S$ *a semantic transition system* $T' \in \wp$ *describing the semantics of the action labeling the entry transition,*
$Act \in$ {*Create*, *Withdraw*, *Satisfy*, *Accept-content*, *Refuse-content*, *Challenge-content*, *Justify-content*, *Defend-content*, *Attack-content*} *is the set of actions,*
$\rightarrow \subseteq S \times Act \times S$ *is the transition relation,*
s_0 *is the start state with* $L(s_0) = \varepsilon$ (*i.e. there is no semantic transition system in* s_0).

The transitions are labeled by the actions applied to SC and to SC contents and the argumentation actions. We write $s \rightarrow s'$ instead of $<s, \bullet, s'> \in \rightarrow$ where $s, s' \in S$ and $\bullet \in Act$. Figure 8.5 illustrates a part of a transition system for a dialogue game protocol.

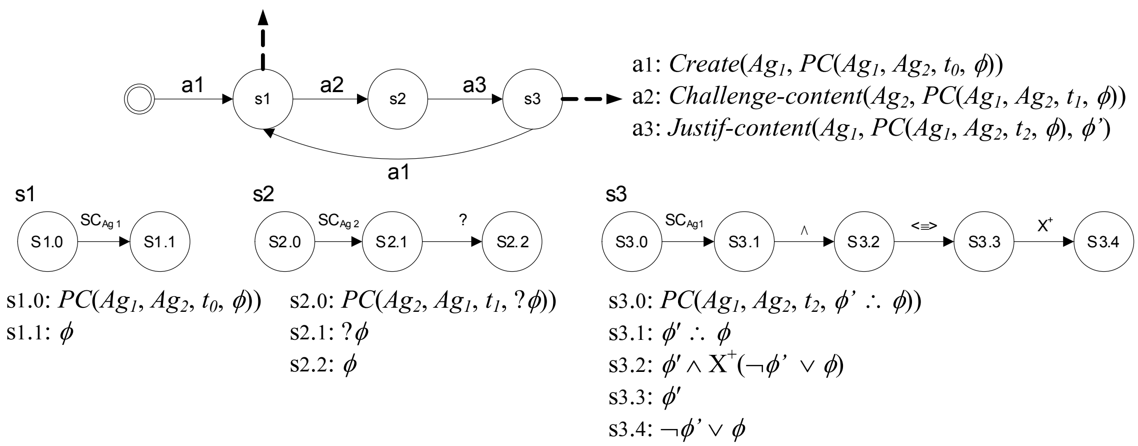


Figure 8.5. A part of a transition system for a dialogue game protocol

8.4.2 Logical Properties

The properties to be verified in the dialogue game protocols specified by DCTL*$_{CAN}$ are action and temporal properties. For example, we can verify if a model of dialogue game protocol satisfies the following property:

$$AG^{+}\mathbf{(}\mathit{Challenge\text{-}content}(Ag_2, PC(Ag_1, Ag_2, t, \phi)) \Rightarrow F^{+}\mathit{Justify\text{-}content}(Ag_1, PC(Ag_1, Ag_2, t, \phi))\mathbf{)}$$

This property indicates that if an agent Ag_2 challenges the content of an Ag_1's propositional commitment (PC), then Ag_1 will justify this content.

Another property capturing the deontic notion of SC is given by the following formula:

$$AG^{+}\mathbf{(}\mathit{Attack\text{-}content}(Ag_2, PC(Ag_1, Ag_2, t, \phi), \phi')) \Rightarrow \mathbf{(}F^{+}\mathit{Defend\text{-}content}(Ag_1, PC(Ag_1, Ag_2, t, \phi)) \vee F^{+}\mathit{Attack\text{-}content}(Ag_1, PC(Ag_2, Ag_1, t', \phi'))\mathbf{)} \vee F^{+}\mathit{Accept\text{-}content}(Ag_1, PC(Ag_2, Ag_1, t', \phi'))\mathbf{)}$$

Thus, we can verify if a model of a dialogue game protocol satisfies the fact that if an agent Ag_2 attacks the content of an agent Ag_1's propositional commitment PC, then Ag_1 will defend its propositional commitment content, attack the Ag_2's argument or accept it.

8.5 Verification of Dialogue Game Protocols

In this section, we use a combination of an automata-theoretic approach and a tableau-based approach to model-checking for our commitment and argument logic.

8.5.1 Alternating Büchi Tableau Automata for DCTL*$_{CAN}$

As a kind of Büchi automata, ABTAs (Bhat, 1998), (Bhat and Cleaveland, 2001) are used in order to prove properties of *infinite* behavior. These automata can be used as an intermediate representation for system properties. Let £ be the set of atomic propositions and let $\Re$ be a set of tableau rule labels defined as follows:

$\Re = \{\wedge, \vee, \neg, ?\} \cup \Re_{Act} \cup \Re_{\neg Act} \cup \Re_{SC} \cup \Re_{Set}$ where $\Re_{Act}$, $\Re_{SC}$ and $\Re_{Set}$ are defined as follows:

$\Re_{Act} = \{<\alpha_{\varphi}>, <\alpha_{SC}>, <C>, <W>, <S^{Ag}_{PC}>, <S^{Ag}_{AC}>, <V^{Ag}_{PC}>, <V^{Ag}_{AC}>, <Rea>, <Ch>, <Acc>, <Ref>, <Jus>, <Att>, <Def>\}$.

$\Re_{SC} = \{[PC_{Ag}], [AC_{Ag}]\}$.

$\Re_{Set} = \{<\equiv>, X^{+}, X^{-}\}$.

The associated tableau rules are given in Figures 8.1, 8.2, 8.3 and 8.4.

Formally, we define ABTAs for our DCTL*$_{CAN}$ logic as follows:

Definition 8.3 *An ABTA for DCTL*$_{CAN}$ *is a 5-tuple* $<Q, l, \rightarrow, q_0, F>$*, where:*
Q is a finite set of states,
$l: Q \rightarrow £ \cup \Re$ *is the state labeling,*
$\rightarrow \subseteq Q \times Q$ *is the transition relation,*
q_0 *is the start state,*
$F \subseteq 2^{Q}$ *is the acceptance condition.*

ABTAs allow us to encode "*top-down proofs*" for temporal formulae. Indeed, an ABTA encodes a proof schema in order to prove, in a goal-directed manner, that a transition system satisfies a temporal formula. Let us consider the following example. We would like to prove that a state *s* in a transition system satisfies a temporal formula of the form $F1 \wedge F2$, where *F1* and *F2* are two formulae. Regardless of the structure of the system, there would be two sub-goals if we want to prove this in a top-down, goal-directed manner. The first would be to prove that *s* satisfies *F1*, and the second would be to prove that *s* satisfies *F2*. Intuitively, an ABTA for $F1 \wedge F2$ would encode this "proof structure" using states for the formulae $F1 \wedge F2$, *F1*, and *F2*. A transition from $F1 \wedge F2$ to each of *F1* and *F2* should be added to the ABTA and the labeling of the state for $F1 \wedge F2$ being "$\wedge$" which is the label of a certain rule. Indeed, in an ABTA, we can consider that: 1) states correspond to "formulae", 2) the labeling of a state is the "logical operator" used to construct the formula, and 3) the transition relation represents a "sub-goal" relationship.

In order to decide about the satisfaction of formulae, we use the notion of the accepting runs of an ABTA on a transition system. These runs are not considered to be finite, but rather infinite, while cycling infinitely many times through acceptance states. In order to define this notion of the ABTA's run, we need to introduce three types of nodes: *positive*, *negative* and *neutral* (neither positive nor negative). Intuitively, nodes classified positive are nodes that correspond to a formula without negation (for example $Create(Ag_1, PC(Ag_1, Ag_2, t, \phi))$), and negative nodes are nodes that correspond to a formula with negation (for example $\neg Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \phi))$). Neutral nodes are used in order to verify the

semantics of an action formula ($act \in Act$) written in the formula to be verified under the form $\neg act$. From the syntax point of view, $\neg act$ means that the action *act* is not performed. For example, if in the formula to be verified appears the sub-formula:
"$\neg$*Justify-content*(Ag_1, $PC(Ag_1, Ag_2, t, \phi)$, ϕ')",
we use in the ABTA neutral nodes in order to verify the semantics of:
"*Justify-content*(Ag_1, $PC(Ag_1, Ag_2, t, \phi)$, ϕ')".
The reason is that in transition systems, and consequently in the sub-transition systems, we have only action formulae without negation, whereas in the formula to be verified, we can have action formulae with negation. We note that we can not use here negative nodes because we do not interested in the formula in itself (i.e. in the example "$\neg$*Justify-content*(Ag_1, $PC(Ag_1, Ag_2, t, \phi)$, ϕ')") but in the semantics of the underlying action (i.e. "*Justify-content*(Ag_1, $PC(Ag_1, Ag_2, t, \phi)$, ϕ')"). In other words, we are not interested in the semantics of the negation action, but in the semantics of the action itself. Section 8.5.5 presents an example (Example 2) illustrating this case. We note here that in order to verify that an action formula $\neg act$ is satisfied, we have to verify that from a given state there is no transition in the transition system labeled by *act*. Definition 8.4 gives the definition of this notion of run. In this definition, elements of the set S of states are denoted s_i or t_i. The explanation of the different closes is given after the definition and a detailed example of this notion of run is given in Figure 8.11 at the end of this chapter.

Definition 8.4 *A run of an ABTA* $B = <Q, l, \rightarrow, q_o, F>$ *on a transition system* $T = <S, Lab, \wp, L, Act, \rightarrow, s_0>$ *is a graph in which the nodes are classified as positive, negative or neutral and are labeled by elements of* $Q \times S$ *as follows*:

1. *The root of the graph is a positive node and is labeled by* $<q_0, s_0>$.
2. *If* φ *is a positive node with label* $<q, s_i>$ *such that* $l(q) = \neg$ *and* $q \rightarrow q'$, *then* φ *has one negative successor labeled* $<q', s_i>$ *and vice versa.*

- *Otherwise, for a positive node* φ *labeled by* $<q, s_i>$:
    3. *If* $l(q) \in £$ *then* φ *is a leaf.*
    4. *If* $l(q) \in \{\wedge, <\equiv>\}$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, \ldots, q_m\}$, *then* φ *has positive successors* $\varphi_1, \ldots, \varphi_m$ *with* φ_j *labeled by* $<q_j, s_i>$ $(1 \leq j \leq m)$.
    5. *If* $l(q) = \vee$ *then* φ *has one positive successor* φ' *labeled by* $<q', s_i>$ *for some* $q' \in \{q' \mid q \rightarrow q'\}$.
    6. *If* $l(q) = X^+$ *and* $q \rightarrow q'$ *and* $\{s' \mid s_i \rightarrow^{\bullet} s'\} = \{t_1, \ldots, t_m\}$ *where* $\bullet \in Act$, *then* φ *has positive successors* $\varphi_1, \ldots, \varphi_m$ *with* φ_j *labeled by* $<q', t_j>$ $(1 \leq j \leq m)$.
    7. *If* $l(q) = X^-$ *and* $q \rightarrow q'$ *and* $\{s' \mid s' \rightarrow^{\bullet} s_i\} = \{t_1, \ldots, t_m\}$ *where* $\bullet \in Act$, *then* φ *has positive successors* $\varphi_1, \ldots, \varphi_m$ *with* φ_j *labeled by* $<q', t_j>$ $(1 \leq j \leq m)$.
    8. *If* $l(q) = <\bullet>$ *where* $\bullet \in Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\bullet} s_{i+1}$ *then* φ *has one positive successor* φ' *labeled by* $<q', s_{i+1,0}>$ *where* $s_{i+1,0}$ *is the initial state of the semantic transition system of* s_{i+1}.
    9. *If* $l(q) = <\bullet>$ *where* $\bullet \in \neg Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\neg\bullet} s_{i+1}$ *then* φ *has one neutral successor* φ' *labeled by* $<q', s_{i+1,0}>$ *where* $s_{i+1,0}$ *is the initial state of the semantic transition system of* s_{i+1}.
    10. *If* $l(q) = <\bullet>$ *where* $\bullet \in \neg Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\bullet'} s_{i+1}$ *where* $\bullet \neq \bullet'$ *and* $\bullet' \in Act$, *then* φ *has one positive successor* φ' *labeled by* $<q', s_{i+1}>$.

- *Otherwise, for a negative node* φ *labeled by* $\langle q, s_i \rangle$:
  11. *If* $l(q) \in$ £ *then* φ *is a leaf.*
  12. *If* $l(q) \in \{\vee, \langle\equiv\rangle\}$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, \ldots, q_m\}$, *then* φ *has negative successors* $\varphi_1, \ldots, \varphi_m$ *with* φ_j *labeled by* $\langle q_j, s_i \rangle$ $(1 \leq j \leq m)$.
  13. *If* $l(q) = \wedge$ *then* φ *has one negative successor* φ' *labeled by* $\langle q', s_i \rangle$ *for some* $q' \in \{q' \mid q \rightarrow q'\}$.
  14. *If* $l(q) = X^+$ *and* $q \rightarrow q'$ *and* $\{s' \mid s_i \rightarrow^{\bullet} s'\} = \{t_1, \ldots, t_m\}$ *where* $\bullet \in Act$, *then* φ *has negative successors* $\varphi_1, \ldots, \varphi m$ *with* φ_j *labeled by* $\langle q', t_j \rangle$ $(1 \leq j \leq m)$.
  15. *If* $l(q) = X^-$ *and* $q \rightarrow q'$ *and* $\{s' \mid s' \rightarrow^{\bullet} s_i\} = \{t_1, \ldots, t_m\}$ *where* $\bullet \in Act$, *then* φ *has negative successors* $\varphi_1, \ldots, \varphi_m$ *with* φ_j *labeled by* $\langle q', t_j \rangle$ $(1 \leq j \leq m)$.
  16. *If* $l(q) = \langle\bullet\rangle$ *where* $\bullet \in Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\bullet} s_{i+1}$ *then* φ *has one negative successor* φ' *labeled by* $\langle q', s_{i+1,0} \rangle$ *where* $s_{i+1,0}$ *is the initial state of the semantic transition system of* s_{i+1}.
  17. *If* $l(q) = \langle\bullet\rangle$ *where* $\bullet \in \neg Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\neg\bullet} s_{i+1}$ *then* φ *has one neutral successor* φ' *labeled by* $\langle q', s_{i+1,0} \rangle$ *where* $s_{i+1,0}$ *is the initial state of the semantic transition system of* s_{i+1}.
  18. *If* $l(q) = \langle\bullet\rangle$ *where* $\bullet \in \neg Act$ *and* $q \rightarrow q'$, *and* $s_i \rightarrow^{\bullet'} s_{i+1}$ *where* $\bullet \neq \bullet'$ *and* $\bullet' \in Act$, *then* φ *has one negative successor* φ' *labeled by* $\langle q', s_{i+1} \rangle$.
- *Otherwise, for a neutral node* φ *labeled by* $\langle q, s_{i,j} \rangle$:
  19. *If* $l(q) = \langle\equiv\rangle$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, q_2\}$ *such that* q_1 *is a leaf, and* $s_{i,j}$ *has a successor* $s_{i,j+1}$, *then* φ *has one positive leaf successor* φ' *labeled by* $\langle q_1, s_{i,j} \rangle$ *and one neutral successor* φ'' *labeled by* $\langle q_2, s_{i,j+1} \rangle$.
  20. *If* $l(q) = \langle\equiv\rangle$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, q_2\}$ *such that* q_1 *is a leaf, and* $s_{i,j}$ *has no successor, then* φ *has one positive leaf successor labeled by* $\langle q_1, s_{i,j} \rangle$.
- *Otherwise, for a positive (negative) node* φ *labeled by* $\langle q, s_{i,j} \rangle$:
  21. *If* $l(q) = \langle\equiv\rangle$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, q_2\}$ *such that* q_1 *is a leaf, and* $s_{i,j}$ *has a successor* $s_{i,j+1}$, *then* φ *has one positive leaf successor* φ' *labeled by* $\langle q_1, s_{i,j} \rangle$ *and one positive (negative) successor* φ'' *labeled by* $\langle q_2, s_{i,j+1} \rangle$.
  22. *If* $l(q) = \langle\equiv\rangle$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1, q_2\}$ *such that* q_1 *is a leaf, and* $s_{i,j}$ *has no successor, then* φ *has one positive leaf successor* φ' *labeled by* $\langle q_1, s_{i,j} \rangle$ *and one positive (negative) successor* φ'' *labeled by* $\langle q_2, s_i \rangle$.
- *Otherwise, for a positive (negative, neutral) node* φ *labeled by* $\langle q, s_{i,j} \rangle$:
  23. *If* $l(q) \in \{\wedge, \vee, ?, X^+, X^-, [SC_{Ag}]\}$ *where* $SC \in \{PC, AC\}$ *and* $\{q' \mid q \rightarrow q'\} = \{q_1\}$, *and* $s_{i,j} \rightarrow^{r} s_{i,j+1}$ *such that* $r = l(q)$, *then* φ *has one positive (negative, neutral) successor* φ' *labeled by* $\langle q_1, s_{i,j+1} \rangle$.

The notion of run of an ABTA on a transition system is a non-synchronized product graph of the ABTA and the transition system. This run uses the label of nodes in the ABTA ($l(q)$), the transitions in the ABTA ($q \rightarrow q'$), and the transitions in the transition system ($s_i \rightarrow s_j$). The product is not synchronized in the sense that it is possible to use transitions in the ABTA while staying in the same state in the transition system (this is the case for example of the closes 2, 4, and 5).

The second close in the definition says that if we have a positive node φ in the product graph such that the corresponding state in the ABTA is labelled with $\neg$ and we have a transition $q \rightarrow q'$ in this ABTA, then φ has one negative successor labelled with $<q', s_i>$. In this case we use a transition from the ABTA and we stay in the same state of the transition system. In the case of a positive node and if the current state of the ABTA is labelled with $\wedge$, all the transitions of this current state of the ABTA are used (close 4). However, if the current state of the ABTA is labelled with $\vee$, only one arbitrary transition from the ABTA is used (close 5). The intuitive idea is that in the case of $\wedge$, all the sub-formulae must be true in order to decide about the formula of the current node of the ABTA, and in the case of $\vee$ only one sub-formula must be true.

The cases in which a transition of the transition system is used are:
1. The current node of the ABTA is labelled with X^+ (which means a next state in the transition system) or X^- (which means a previous state in the transition system). This is the case of the closes 6, 7, 14, and 15. In this case we use all the transitions from the current state s_i to next or previous states of the transition system.
2. The current state of the ABTA and a transition from the current state of the transition system are labelled with the same action. This is the case of the closes 8 and 16. In this case, the current transition of the ABTA and the transition from the current state s_i of the transition system to a state $s_{i+1,\,0}$ of the associated semantic transition system are used. The idea is to start the parsing of the formula coded in the semantic transition system.
3. The current state of the ABTA and a transition from the current state of the transition system are labelled with the same action which is preceded by $\neg$ in the ABTA. This is the case of the closes 9 and 17. In this case, the current transition of the ABTA and the transition from the current state s_i of the transition system to a state $s_{i+1,\,0}$ of the associated semantic transition system are used. The successor node is classified neutral. This allows us to verify the structure of the formula coded in the transition system.
4. The current state of the ABTA and a transition from the current state of the transition system are labelled with different actions where the state of the ABTA is labelled with a negative formula. This is the case of the closes 10 and 18. In this case, the formula is satisfied, but its structure cannot be verified. Consequently, the current transition of the ABTA and the transition from the current state s_i of the transition system to a next state s_{i+1} are used. This means that, we do not visit the associated semantic transition system.

Finally, the closes 19, 20, 21, 22, and 23 deal with the case of verifying the structure of the commitment formulae in the sub-transition systems. In these closes, transitions $s_{i,j} \rightarrow s_{i,j+1}$ are used. We note here that when $s_{i,j}$ has no successor, the formula contained in this state is an atomic formula or a boolean formula whose all the sub-formulae are atomic (for example $p \wedge q$ where p and q are atomic).

We also need to define the notion of success of a run for the correctness of the model checking. To define this notion, we first introduce the following terminology:

In an ABTA, every infinite path has a suffix that contains either positive or negative nodes, but not both. Such a path is referred to as *positive* in the former case and *negative* in the latter.

Let $p \in \Phi p$ and let s_i be a state in a transition system T. Then $s_i \vDash_T p$ iff $p \in Lab(s_i)$ and $s_i \vDash_T \neg p$ iff $p \notin Lab(s_i)$.

Let $s_{i,j}$ be a state in a semantic transition system of a transition system T. Then $s_{i,j} \vDash_T p$ iff $p \in Lab'(s_{i,j})$ and $s_{i,j} \vDash_T \neg p$ iff $p \notin Lab'(s_{i,j})$.

Definition 8.5 *Let r be a run of ABTA B = <Q, l, →, q_0, F> on a transition system T = <S, Lab, ℘, L, Act, →, s_0>. The run r is successful iff every leaf and every infinite path in r is successful. A successful leaf is defined as follows:*

1- *A positive leaf labeled by <q, s_i> is successful iff* $s_i \vDash_T l(q)$ or $l(q) = <\bullet>$ *where* $\bullet \in Act$ *and there is no s_j such that* $s_i \rightarrow^{\bullet} s_j$.

2- *A positive leaf labeled by <q, $s_{i,j}$> is successful iff* $s_{i,j} \vDash_T l(q)$

3- *A negative leaf labeled by <q, s_i> is successful iff* $s_i \vDash_T \neg l(q)$ or $l(q) = <\bullet>$ *where* $\bullet \in Act$ *and there is no s_j such that* $s_i \rightarrow^{\bullet} s_j$.

4- *A negative leaf labeled by <q, $s_{i,j}$> is successful iff* $s_{i,j} \vDash_T \neg l(q)$

5- *All neutral leaves are not successful.*

A successful infinite path is defined as follows:

1- *A positive path is successful iff* $\forall f \in F$, $\exists q \in f$ *such that q occurs infinitely often in the path. This condition is called the Büchi condition.*

2- *A negative path is successful iff* $\exists f \in F$, $\forall q \in f$, *q does not occur infinitely often in the path. This condition is called the co-Büchi condition.*

We note here that a positive or negative leaf labeled by <q, s> such that $l(q) = <\bullet>$ where $\bullet \in Act$ and there is no s' such that $s \rightarrow^{\bullet} s'$ is considered a successful leaf because we can not consider it unsuccessful. The reason is that it is possible to find a transition labeled by • and starting from another state s'' in the transition system. This is the case of the leaf labeled by (<*Ch*>, s_0) in the Example 2, Section 8.5.5 (see Figure 8.11, Section 8.5.6). If we consider such a leaf unsuccessful, then even if we find a successful infinite path, the run will be considered unsuccessful. However this is false.

An ABTA B accepts a transition system T iff there exists a successful run of B on T.

8.5.2 Translating DCTL*$_{\text{CAN}}$ into ABTA

The procedure for translating a DCTL*$_{\text{CAN}}$ formula $p = E\phi$ to an ABTA B uses goal-directed rules in order to build a tableau from this formula. Indeed, these proof rules are conducted in a top-down fashion in order to determine whether states satisfy properties or not. The tableau is constructed by exhaustively applying the rules contained in Figures 8.1, 8.2, 8.3 and 8.4 to p. Then, B can be extracted from this tableau as follows. First, we generate the states and the transitions. Intuitively, states will correspond to state formulae, with the start state being p. To generate new states from an existing state for a formula p', we determine which rule is applicable to p', starting with $R1$, by comparing the form of p' to the formula appearing in the "goal position" of each rule. Let $rule(q)$ denote the rule applied at node q. The labeling function l of states is defined as follows. If q does not have any successor, then $l(q) \in £$. Otherwise, the successors of q are given by *rule*(q). The label of the rule becomes the label of the state q, and the sub-goals of the rule are then added as states related to q by transitions.

A tableau for a DCTL*$_{CAN}$ formula p is a maximal proof tree having p as its root and constructed using $R1$-$R32$. If p' results from the application of a rule to p, then we say that p' is a child of p in the tableau. The height of a tableau is defined as the length of the longest sequence $<p_0, p_1, \ldots>$, where p_{i+1} is the child of p_i (Cleaveland, 1990). Finally, in order to compute the successful run of the generating ABTA, we should compute the acceptance states F. For this purpose we use the following definition.

Definition 8.6 *Let q be a state in an ABTA B and Q the set of all states. Suppose* $\phi = \phi_1 \, U^+ \, \phi_2 \in q$[7]. *We define the set* F_ϕ *as follows*:
$F_\phi = \{q' \in Q \mid (\phi \notin q' \text{ and } X^+\phi \notin q') \text{ or } \phi_2 \in q'\}$.
The acceptance set F is defined as follows:
$F = \{F_\phi \mid \phi = \phi_1 \, U^+ \, \phi_2 \text{ and } \exists q \in B, \phi \in q\}$.

According to this definition, a state that contains the formula ϕ or the formula $X^+\phi$ is not an acceptance state. The reason is that according to Definition 8.4, there is a transition from a state containing ϕ to a state containing $X^+\phi$ and vice versa. Therefore, according to Definition 8.5, there is a successful run in the ABTA B. However, we can not decide about the satisfaction of a formula using this run. The reason is that in an infinite cycle including a state containing ϕ and a state containing $X^+\phi$, we can not be sure that a state containing ϕ_2 is reachable. However, according to the semantics of U^+, the satisfaction of ϕ needs that a state containing ϕ_2 is reachable while passing by states containing ϕ_1.

8.5.3 Termination

In this section we prove the termination of the translation procedure. Since this procedure is based on tableau rules, we need to prove the finiteness of the tableau. The methodology that we follow is inspired by (Cleaveland, 1990), (Adi et al., 2003).

If σ_2 is a DCTL*$_{CAN}$ formula resulting from the application of a rule to a DCTL*$_{CAN}$ formula σ_1, then we say that σ_2 is a child of σ_1 in the tableau and σ_1 is the parent of σ_2. The *height* of a tableau (Cleaveland, 1990) is defined as the length of the longest sequence $<\sigma_0, \sigma_1, \ldots>$, where σ_i is the parent of σ_{i+1}. To prove the finiteness of a tableau, we will establish that each formula has a maximum height tableau.

Intuitively, to show the finiteness of the tableau, we will define a strict ordering relation $\prec$ between DCTL*$_{CAN}$ formulae and then show that:
1- if σ_1 is the parent of σ_2, then $\sigma_1 \prec \sigma_2$.

[7] Here we consider the until formula because is the formula that allows paths to be infinite.

2- the strict ordering relation $\prec$ has no infinite ascending chains.

The ordering relation $\prec$ should reflect the fact that applying tableau rules results in shorter formulae or recursive formulae. The idea is to prove that the number of nodes of the ABTA is finite. Therefore, the definition of this ordering is based either on the fact that formulae are recursive or on the length of formulae. We notice that in the case of recursive formulae, we obtain cycles which are infinite paths on a finite number of nodes. The length of a formula is defined inductively as follows:

Definition 8.7 *The length of a formula ψ denoted by $|\psi|$ is the number of variables and operators in ψ i.e.*

$|\psi| = 1$ if ψ *is an atomic formula*
$|\neg\psi| = 1 + |\psi|$
$|\psi_1 \wedge \psi_2| = 1 + |\psi_1| + |\psi_2|$
$|\psi_1 \vee \psi_2| = 1 + |\psi_1| + |\psi_2|$
$|?\psi| = 1 + |\psi|$
$|\psi_1 \therefore \psi_2| = 1 + |\psi_1| + |X^+(\neg\psi_1 \vee \psi_2)|$
$|X\psi| = 1 + |\psi|$ *where* $X \in \{X^+, X^-\}$
$|\psi_1 \, U \, \psi_2| = 1 + |\psi_1| + |\psi_2|$ *where* $(U, X) \in \{(U^+, X^+), (U^-, X^-)\}$
$|PC(Ag_1, Ag_2, t, \psi)| = 1 + |\psi|$
$|AC(Ag_1, Ag_2, t, (\alpha)\psi)| = 1 + |\psi|$
$|Perform(\alpha)\psi| = 1 + |\psi|$
$|Perform(\alpha)SC(Ag_1, Ag_2, t, \psi)| = 1 + |SC(Ag_1, Ag_2, t, \psi)|$
$|Create(Ag_1, SC(Ag_1, Ag_2, t, \psi))| = 1 + |SC(Ag_1, Ag_2, t, \psi)|$
$|Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \psi))| = 1 + |\neg SC(Ag_1, Ag_2, t, \psi)|$
$|Satisfy(Ag_1, PC(Ag_1, Ag_2, t, \psi))| = 1 + |\psi|$
$|Satisfy(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi)))| = 1 + |Perform(\alpha)\psi|$
$|Violate(Ag_1, PC(Ag_1, Ag_2, t, \psi))| = 1 + |\neg\psi|$
$|Violate(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi)))| = 1 + |\neg Perform(\alpha)\psi|$
$|Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \psi))| = 1 + |SC(Ag_1, Ag_2, t, \psi)|$
$|Challenge\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, \psi))| = 1 + |PC(Ag_2, Ag_1, t', ?\psi)|$
$|Accept\text{-}content(Ag_2, SC(Ag_1, Ag_2, t, \psi))| = 1 + |SC(Ag_2, Ag_1, t', \psi)|$
$|Refuse\text{-}content(Ag_2, SC(Ag_1, Ag_2, t, \psi))| = 1 + |SC(Ag_2, Ag_1, t', \neg\psi)|$
$|Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi), \psi')| = 1 + |PC(Ag_1, Ag_2, t', \psi' \therefore \psi)|$
$|Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, \psi), \psi')| = 1 + |PC(Ag_2, Ag_1, t', \psi' \therefore \neg\psi)|$
$|Defend\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi), \psi')| = 1 + |PC(Ag_1, Ag_2, t', \psi' \therefore \psi)|$

The ordering relation $\prec$ is defined as follows:

Definition 8.8 *Let* $\sigma_1 = E(\psi_1)$ *and* $\sigma_2 = E(\psi_2)$ *be two* $DCTL^*_{CAN}$ *formulae. Then,* $\sigma_1 \prec \sigma_2$ *holds if*
1- $\sigma_1 \lessdot \sigma_2$

2- *$\sigma_1 \nleqslant \sigma_2$ and* $|\psi_1| > |\psi_2|$.
where $\sigma_1 \leqslant \sigma_2$ iff $X\psi_1$ *appears in* ψ_2

The first close is used when we have a recursive formula (this means that an *until* formula).

$\prec$ is irreflexive, asymmetric and transitive. The proof is straightforward from the definition since $>$ and $\leqslant$ are strict ordering relations.

In what follows, the notation $\sigma_1 \rightarrow_R \sigma_2$ means that σ_1 is the parent of σ_2 using a tableau rule R. Now, let us prove the following lemma.

Lemma 8.9 *Let* $\sigma_1 = \mathrm{E}(\psi_1)$ *and* $\sigma_2 = \mathrm{E}(\psi_2)$ *be two DCTL*$_{CAN}$ formulae. Then:*
$\sigma_1 \rightarrow_R \sigma_2 \Rightarrow \sigma_1 \prec \sigma_2$.

Proof
The proof is based on the analysis of the different cases of our tableau rules. Most cases are straightforward. Here we only consider rules $R7$, $R9$, $R30$, and $R32$.

$R = R7$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \sigma_1 = \mathrm{E}(\Phi, Perform(\alpha)\psi)$, $\sigma_2 = \mathrm{E}(\Phi, \psi)$
$\Rightarrow \sigma_1 \prec \sigma_2$ (from the definition of $\prec$ (Definition 8.8) and the fact that $|Perform(\alpha)\psi| = 1 + |\psi| > |\psi|$)

$R = R9$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \sigma_1 = \mathrm{E}(\Phi, Create(Ag_1, SC(Ag_1, Ag_2, t, \psi)$, $\sigma_2 = \mathrm{E}(\Phi, SC(Ag_1, Ag_2, t, \psi))$
$\Rightarrow \sigma_1 \prec \sigma_2$ (from the definition of $\prec$ and the fact that $|Create(SC(Ag_1, Ag_2, t, \psi)| = 1 + |SC(Ag_1, Ag_2, t, \psi)| > |SC(Ag_1, Ag_2, t, \psi)|$)

$R = R30$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \sigma_1 = \mathrm{E}(\Phi, \psi_1 \therefore \psi_2)$, $\sigma_2 = \mathrm{E}(\Phi, \psi_1, X^+(\neg\psi_1 \vee \psi_2))$
$\Rightarrow \sigma_1 \prec \sigma_2$ (from the definition of $\prec$ and the fact that $|\psi_1 \therefore \psi_2| = 1 + |\psi_1| + |X^+(\neg\psi_1 \vee \psi_2)|$

$R = R32$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \sigma_1 = \mathrm{E}(\Phi, \psi_1\, U^+ \psi_2)$, $\sigma_2 = \mathrm{E}(\Phi, \psi_2)$ or $\mathrm{E}(\Phi, \psi_1, X^+(\psi_1\, U^+ \psi_2))$
$\Rightarrow \sigma_1 \prec \sigma_2$ (from the definition of $\prec$ and the fact that $\sigma_1 \leqslant \sigma_2$
□

To show that the ordering relation has no infinite ascending chains, we use the notion of Fischer-Ladner closure of a formula ψ ($CL(\psi)$) (Emerson et al., 1993). The idea underlying the definition of this notion is to prove that if a tableau has a root ψ, then all formulae ψ' of this tableau have a formula in $CL(\psi)$ (i.e. $\psi' \in CL(\psi)$). Furthermore, if we prove that $CL(\psi)$ is a finite set, then we conclude that each formula appearing in a given tableau belongs to a finite set. This result will be very helpful to prove that the ordering relation $\prec$ has no infinite ascending chains.

Definition 8.10 *Let ψ be a DCTL*$_{CAN}$ formula. The Fischer-Ladner closure of ψ, $CL(\psi)$ is the smallest set such that the following hold:*
If ψ is an atomic formula then $\{\psi\} \subseteq CL(\psi)$
If $\psi = \neg\psi_1$ *then* $CL(\psi_1) \subseteq CL(\psi)$ and $\{\neg\psi_1\} \subseteq CL(\psi)$
If $\psi = \psi_1 \wedge \psi_2$ *then* $CL(\psi_1) \subseteq CL(\psi)$ *and* $CL(\psi_2) \subseteq CL(\psi)$ *and* $\{\psi_1 \wedge \psi_2\} \subseteq CL(\psi)$
If $\psi = \psi_1 \vee \psi_2$ *then* $CL(\psi_1) \subseteq CL(\psi)$ and $CL(\psi_2) \subseteq CL(\psi)$ *and* $\{\psi_1 \vee \psi_2\} \subseteq CL(\psi)$
If $\psi = ?\psi_1$ *then* $CL(\psi_1) \subseteq CL(\psi)$ *and* $\{?\psi_1\} \subseteq CL(\psi)$
If $\psi = \psi_1 \therefore \psi_2$ *then*
$CL(\psi_1) \subseteq CL(\psi)$ *and* $CL(X^+(\neg\psi_1 \vee \psi_2)) \subseteq CL(\psi)$ *and* $\{\psi_1 \therefore \psi_2\} \subseteq CL(\psi)$
If $\psi = X\psi_1$ *then* $CL(\psi_1) \subseteq CL(\psi)$ *and* $\{X\psi_1\} \subseteq CL(\psi)$ *where* $X \in \{X^+, X^-\}$
If $\psi = \psi_1 U \psi_2$ *then*
$CL(\psi_1) \subseteq CL(\psi)$ *and* $CL(\psi_2) \subseteq CL(\psi)$ *and* $CL(X(\psi_1 U \psi_2)) \subseteq CL(\psi)$ *and* $\{\psi_1 U \psi_2\} \subseteq CL(\psi)$ *where* $(U, X) \in \{(U^+, X^+), (U^-, X^-)\}$
If $\psi = SC(Ag_1, Ag_2, t, \psi_1)$ *then* $CL(\psi_1) \subseteq CL(\psi)$ *and* $\{SC(Ag_1, Ag_2, t, \psi_1)\} \subseteq CL(\psi)$
If $\psi = AC(Ag_1, Ag_2, t, (\alpha, \psi_1))$ *then*
$CL(\psi_1) \subseteq CL(\psi)$ *and* $\{AC(Ag_1, Ag_2, t, (\alpha, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Perform(\alpha)\psi_1$ *then* $CL(\psi_1) \subseteq CL(\psi)$ *and* $\{Perform(\alpha)\psi_1\} \subseteq CL(\psi)$
If $\psi = Perform(\alpha)SC(Ag_1, Ag_2, t, \psi_1)$ *then*
$CL(SC(Ag_1, Ag_2, t, \psi_1)) \subseteq CL(\psi)$ *and* $\{Perform(\alpha)SC(Ag_1, Ag_2, t, \psi_1)\} \subseteq CL(\psi)$
If $\psi = Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(SC(Ag_1, Ag_2, t, \psi_1)) \subseteq CL(\psi)$ *and* $\{Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(\neg SC(Ag_1, Ag_2, t, \psi_1)) \subseteq CL(\psi)$ *and* $\{Withdraw(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Satisfy(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(\psi_1) \subseteq CL(\psi)$ *and* $\{Satisfy(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Satisfy(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi_1)))$ *then*
$CL(Perform(\alpha)\psi_1) \subseteq CL(\psi)$ *and* $\{Satisfy(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi_1)))\} \subseteq CL(\psi)$
If $\psi = Violate(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(\neg\psi_1) \subseteq CL(\psi)$ *and* $\{Violate(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Violate(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi_1)))$ *then*
$CL(\neg Perform(\alpha)\psi_1) \subseteq CL(\psi)$ *and* $\{Violate(Ag_1, AC(Ag_1, Ag_2, t, (\alpha, \psi_1)))\} \subseteq CL(\psi)$
If $\psi = Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(SC(Ag_1, Ag_2, t, \psi_1)) \subseteq CL(\psi)$ *and* $\{Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Challenge\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))$ *then*
$CL(PC(Ag_1, Ag_2, t', ?\psi_1)) \subseteq CL(\psi)$

$\quad$ *and* $\{Challenge\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Accept\text{-}content(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$ *then*
$\quad CL(SC(Ag_1, Ag_2, t', \psi_1)) \subseteq CL(\psi)$
$\quad$ *and* $\{Accept\text{-}content(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Refuse\text{-}content(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$ *then*
$\quad CL(SC(Ag_1, Ag_2, t', \neg\psi_1)) \subseteq CL(\psi)$
$\quad$ *and* $\{Refuse\text{-}content(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))\} \subseteq CL(\psi)$
If $\psi = Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)$ *then*
$\quad CL(PC(Ag_1, Ag_2, t', \psi_2 \therefore \psi_1)) \subseteq CL(\psi)$
$\quad$ *and* $\{Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)\} \subseteq CL(\psi)$
If $\psi = Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)$ *then*
$\quad CL(PC(Ag_2, Ag_1, t', \psi_2 \therefore \neg\psi_1)) \subseteq CL(\psi)$
$\quad$ *and* $\{Attack\text{-}content(Ag_2, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)\} \subseteq CL(\psi)$
If $\psi = Defend\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)$ *then*
$\quad CL(PC(Ag_1, Ag_2, t', \psi_2 \therefore \psi_1)) \subseteq CL(\psi)$
$\quad$ *and* $\{Defend\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \psi_1), \psi_2)\} \subseteq CL(\psi)$

Lemma 8.11 *Let* ψ be a formula, then $CL(\psi)$ is finite and bounded in size by $2|\psi|$.

Proof
The proof is based on the induction of the structure of ψ. Most cases are straightforward. Here we only consider the four following cases:

1- $\psi = X\psi_1$, where $X \in \{X^+, X^-\}$.
We have:
$CL(X\psi_1) = \{X\psi_1\} \cup CL(\psi_1)$
Therefore:
$|CL(X\psi_1)| = 1 + |CL(\psi_1)|$
Then, by using the induction hypothesis, we conclude that:
$|CL(X\psi_1)| \leq 1 + 2|\psi_1| \leq 2(1 + |\psi_1|)$
Then, by using Definition 8.7 we obtain:
$|CL(X\psi_1)| \leq 2|X\psi_1|$

2- $\psi = \psi_1 \text{ U } \psi_2$, where $U \in \{U^+, U^-\}$.

We have:
$CL(\psi_1 \text{ U } \psi_2) = \{\psi_1 \text{ U } \psi_2\} \cup CL(\psi_1) \cup CL(\psi_2) \cup CL(X(\psi_1 \text{ U } \psi_2))$
$\qquad = \{\psi_1 \text{ U } \psi_2\} \cup CL(\psi_1) \cup CL(\psi_2) \cup \{X(\psi_1 \text{ U } \psi_2)\}$
Therefore:
$|CL(\psi_1 \text{ U } \psi_2)| = 2 + |CL(\psi_1)| + |CL(\psi_2)|$
Then, by using the induction hypothesis and the previous case, we conclude that:
$|CL(\psi_1 \text{ U } \psi_2)| \leq 2 + 2|\psi_1| + 2|\psi_2| + |X(\psi_1 \text{ U } \psi_2)|$
Then, by using Definition 8.7 we obtain:
$|CL(\psi_1 \text{ U } \psi_2)| \leq 2|\psi_1 \text{ U } \psi_2|$

3- $\psi = SC(Ag_1, Ag_2, t, \psi_1)$

We have:
$CL(SC(Ag_1, Ag_2, t, \psi_1)) = \{SC(Ag_1, Ag_2, t, \psi_1)\} \cup CL(\psi_1)$
Therefore:
$|CL(SC(Ag_1, Ag_2, t, \psi_1))| = 1 + |CL(\psi_1)|$
Then, by using the induction hypothesis, we conclude that:
$|CL(SC(Ag_1, Ag_2, t, \psi_1))| \leq 1 + 2|\psi_1|$
Then, by using Definition 8.7 we obtain:
$|CL(SC(Ag_1, Ag_2, t, \psi_1))| \leq 2|SC(Ag_1, Ag_2, t, \psi_1)|$

4- $\psi = Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))$

We have:
$CL(Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))) = \{Create(SC(Ag_1, Ag_2, t, \psi_1))\}$
$\cup\ CL(SC(Ag_1, Ag_2, t, \psi_1))$
Therefore:
$|CL(Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1)))| = 1 + 2|CL(SC(Ag_1, Ag_2, t, \psi_1))|$
Then, by using the previous case, we conclude that:
$|CL(Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))| \leq 1 + 2|SC(Ag_1, Ag_2, t, \psi_1)|$
Then, by using Definition 8.7 we obtain:
$|CL(Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1)))| \leq 2|Create(Ag_1, SC(Ag_1, Ag_2, t, \psi_1))|$
□

The next lemma establishes the link between tableau rules and Fischer-Ladner closure of formulae.

Lemma 8.12 *Let* $\sigma_1 = E(\Phi, \psi_1)$ *and* $\sigma_2 = E(\Phi, \psi_2)$ *be two* $DCTL^*_{CAN}$ *formulae. Then:*
$\sigma_1 \rightarrow_R \sigma_2 \Rightarrow CL(\psi_2) \subseteq CL(\psi_1)$.

Proof
The proof is based on the case analysis of the rule R. Most cases are straightforward. Here we consider the rules $R7$, $R9$, $R30$, and $R32$.

$R = R7$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \psi_1 = Perform(\alpha)\psi_2$
$\Rightarrow$ (Definition of $CL(Perform(\alpha)\psi)$)
$CL(\psi_2) \subseteq CL(\psi_1)$

$R = R9$:

$\sigma_1 \rightarrow_R \sigma_2$
$\Rightarrow \psi_1 = Create(\psi_2)$
$\Rightarrow$ (Definition of $CL(Create(\psi_2))$
$CL(\psi_2) \subseteq CL(\psi_1)$

$R = R30$:

$\sigma_1 \rightarrow_R \sigma_2$

$\Rightarrow E(\Phi, \psi_1) = E(\Phi, \psi \therefore \psi')$, $E(\Phi, \psi_2) = E(\Phi, \psi, X^+(\neg\psi \vee \psi'))$

$\Rightarrow CL(\psi_1) = \{\psi \therefore \psi'\} \cup CL(\psi) \cup CL(X^+(\neg\psi \vee \psi'))$

$\Rightarrow CL(\psi_2) \subseteq CL(\psi_1)$

$R = R32$:

$\sigma_1 \rightarrow_R \sigma_2$

$\Rightarrow E(\psi_1) = E(\psi U^+ \psi')$, $E(\psi_2) = E(\psi')$ or $E(\psi, X^+(\psi U^+ \psi'))$

$\Rightarrow CL(\psi_1) = \{\psi U^+ \psi'\} \cup CL(\psi) \cup CL(\psi') \cup CL(X^+(\psi U^+ \psi'))$

$\Rightarrow CL(\psi_2) \subseteq CL(\psi_1)$

□

Intuitively, $\sigma_i \prec \sigma_j$ holds if σ_i is an ancestor of σ_j in some tableau, i.e. if there are rules Ri, ..., Rj such that: $\sigma_i \rightarrow_{Ri} \sigma_{i+1} \ldots \rightarrow_{Rj} \sigma_j$

Lemma 8.13 *The ordering relation $\prec$ has no infinite ascending chains.*

Proof

Suppose that there exists an infinite chain: $\sigma_1 \prec \sigma_2 \prec \ldots$

From Lemma 8.12, it follows that $CL(\psi_i) \subseteq CL(\psi_{i-1}) \subseteq \ldots CL(\psi_1)$

Since $CL(\psi_1)$ is finite (from Lemma 8.11), it follows that:

$\exists j, \forall k \geq j, CL(\psi_k) = CL(\psi_j)$ with $\sigma_j \prec \sigma_{j+1} \prec \ldots \sigma_k \prec \ldots$

However, this is contradictory (from Lemma 8.12).

□

Now, we can easily prove the finiteness theorem as shown below.

Theorem 8.14 *For any DCTL*$^*_{CAN}$ *formula σ_1, there is a maximum height tableau has σ_1 as a root.*

Proof

Suppose that there exists a tableau with root σ_1 having an infinite path:

$\sigma_1 \rightarrow_{Ri} \sigma_2 \rightarrow_{Rj} \sigma_3 \ldots$

where Ri, Rj, ... $\in \{R1, \ldots, R32\}$. Then, from Lemma 8.9 and from the fact that the ordering relation $\prec$ is transitive (since $<$ is transitive), it follows that there exists an infinite chain:

$\sigma_1 \prec \sigma_2 \prec \ldots$

However this is contradictory from Lemma 8.13.

□

8.5.4 Soundness and Completeness

Soundness and completeness of our method are stated by the following theorem.

Theorem 8.15 *Let ψ be a DCTL*$_{CAN}$ formula and B_ψ the ABTA obtained by the translation procedure described above, and let $T = <S, \wp, L, Act, \rightarrow, s_0>$ be a transition system that represents a dialogue game protocol. Then $s_0 \models_T \psi$ iff T is accepted by B_ψ.*

Proof

This theorem is a consequence of Proposition 8.16 and Lemmas 8.19, 8.20 and 8.21.

Proposition 8.16 *Let r a run of an ABTA B on a transition system T. In all infinite paths of r, the semantics of the action formulae appearing in these paths is verified.*

Proof

The proof follows from Definitions 8.4 and 8.5. Indeed, the only case in which the semantics of an action formula is not respected is the case of a positive leaf $<q, s_{i,j}>$ such that $s_{i,j} \models_T \neg l(q)$. Because infinite paths do not encounter any leaf, the semantics of these formulae is verified in these paths.

□

Now, we introduce the following definitions:

Definition 8.17 *Let φ_i a node in the run r of B_ψ labeled by $<q, s>$ and let $\sigma = <\varphi_0, \ldots>$ be an infinite path in the run r. Let $q_0, q_1, \ldots$ be the corresponding sequence of B_ψ states. σ is said to be successful iff for every formula $\phi \equiv \phi_1 \mathrm{U}^+ \phi_2 \in q_i$ there exists $j \geq i$ such that $\phi_2 \in q_j$.*

Definition 8.18 *Let $\sigma_T = <s_0, s_1, \ldots>$ be a path in T, such that $<s_i, s_j> \in \sigma_T$ iff $<(q_l, s_i), (q_m, s_j)> \in \sigma$ or $<(q_l, s_i), (q_m, s_{j,0})> \in \sigma$ for some l and m. In addition, let $\sigma_{T'} = <s_{i,0}, s_{i,1}, \ldots>$ be a path in a semantic transition system T' of T. If $<(q_l, s_i), (q_m, s_{j,0})> \in \sigma$ then $<s_{j,j'}, s_{j,j'+1}> \in \sigma_{T'}$* iff $<(q_l, s_{j,j'}), (q_m, s_{j,j'+1})> \in \sigma$.

We note that if we have a run r in which a leaf $<q, s_{i,j}>$ is unsuccessful, then we conclude that the semantics is not respected and consequently the property to be verified is not satisfied. However, if r contains an unsuccessful leaf $<q, s_i>$, we conclude only that the property is not satisfied (there is no need to verify the semantics).

Lemma 8.19 *Let ψ be a DCTL*$_{CAN}$ state formula and $T = <S, \wp, L, Act, \rightarrow, s_0>$ be a transition system such that $s_0 \models_T \psi$. Also let B_ψ the corresponding ABTA. Then T is accepted by B_ψ.*

Proof

To prove that T is accepted by B_ψ, we have to prove that there exists a run r of B_ψ on T such that all leaves and all infinite paths in the run are successful.

Let us assume that $s_0 \models_T \psi$. First, let us suppose that there exists a leaf $<q, s>$ in r such that $s \models \neg l(q)$. Since the application of tableau rules does not change the satisfaction of formulae, it follows from the definition of r that $s_0 \models_T \neg\psi$ which contradicts our assumption.

Now, we will prove that all infinite paths are successful. The proof proceeds by contradiction. ψ is a state formula that we can write under the form $E\Phi$, where Φ is a set of path formulae. Let us assume that there exists an unsuccessful infinite path σ in r and prove that $\sigma_T \models_T \neg\Phi$. The fact that σ is infinite implies that $R32$ occurs at infinitely many position in σ and that ϕ_1 U^+ $\phi_2 \subseteq \Phi$. Since σ is unsuccessful, there is a formula ϕ_1 U^+ $\phi_2 \in q_i$ such that for all $j \geq i$ we have $\phi_2 \notin q_j$. When this formula appears in the ABTA at the position q_i, we have $l(q_i) = \vee$. Thus, according to the definition of r and the form of $R32$, the current node φ_1 of r labeled by $<q_i, s>$ has one successor φ_2 labeled by $<q_{i+1}, s>$ with ϕ_1 U^+ $\phi_2 \in q_i$ and $\{\phi_1, X^+(\phi_1\ U^+\ \phi_2)\} \subseteq q_{i+1}$. Therefore, $l(q_{i+1}) = \wedge$, and φ_2 has a successor φ_3 labeled by $<q_{i+2}, s>$ with $X^+(\phi_1\ U^+\ \phi_2) \in q_{i+2}$. Using $R29$ and the fact that $l(q_{i+2}) = X^+$, the successor φ_4 of φ_3 is labeled by $<q_{i+3}, s'>$ with ϕ_1 U^+ $\phi_2 \in q_{i+3}$ and $s \to s'$. This process will be repeated infinitely since the path is unsuccessful. It follows that there is no s in T such that $s \models_T \phi_2$. Thus, according to the semantics of ϕ_1 U^+ ϕ_2, there is no s in T such that $s \models_T \phi_1$ U^+ ϕ_2. Therefore, $\sigma_T \models_T \neg\Phi$.
☐

Lemma 8.20 *Let ψ be a DCTL*$_{CAN}$ state formula and B_ψ the corresponding ABTA, and let $T = <S, \wp, L, Act, \to, s_0>$ be a transition system such that T is accepted by B_ψ. Then $s_0 \models_T \psi$.*

Proof

The proof proceeds by contradiction. We assume that $s_0 \models_T \neg\psi$ and we prove that r contains a failed path such that one of the following holds: either σ (a path in the run r of B_ψ on T) is finite and the leaf is unsuccessful or σ is infinite and unsuccessful. Since $s_0 \models_T \neg\psi$ there is a path Π_T in T such that $\Pi_T \models_T \neg\phi$ *for* $\phi \in \Phi$ or there is a path $\Pi_{T'}$ in a semantic transition system T' of T such that $\Pi_{T'} \models_T \neg\phi$ *for* $\phi \in \Phi$. The idea is to show that r contains a failed path σ such that:

1. σ_T is a prefix of Π_T or $\sigma_{T'}$ is a prefix of $\Pi_{T'}$ and
2. if $\sigma_T = <s_{\varphi 0}, \ldots, s_{\varphi i}>$, then for all $\phi \in \varphi_i$, we have $\Pi_T(s_{\varphi i}) \models_T \neg\phi$ or there is a sub-state $s'_{\varphi i}$ of $s_{\varphi i}$ such that $\Pi_{T'}(s'_{\varphi i}) \models_T \neg\phi$ where σ_T ($\sigma_{T'}$) is a path in T (in T') constructed from σ as explained in Definition 8.18 and $s_{\varphi i}$ is the state that correspond to the node φ_i.

We proceed by an inductive construction of σ. For $|\sigma| = 1$, we have $\sigma = <\varphi_0>$ and $\sigma_T = <s_0>$. Thus, σ_T is a prefix of Π_T and $\Pi_T(0) \models_T \neg\phi$ since $\Pi_T(0) = \Pi_T$. First, we suppose that σ is finite. Using the construction process of a run, we can construct such a path from $\Pi_T = <s_0, \ldots, s_n>$ and eventually from $\Pi_{T'} = <s_{n,0}, \ldots, s_{n,n'}>$ such that $\sigma = <(s_0, q_0) \ldots, (s_n, q_m)>$ or $\sigma = <(s_0, q_0) \ldots, (s_{n,n'}, q_m)>$. Since $l(q_m)$ is a sub formula of ϕ obtained by using some tableau rules, and $\Pi_T \models_T \neg\phi$ or $\Pi_{T'} \models_T \neg\phi$, it follows that

$s_n \vDash_T \neg l(q_m)$ or $s_{n,n'} \vDash_T \neg l(q_m)$. Therefore, σ is a failed path. Now, we assume that we have constructed σ so that $|\sigma| = i+1$, for some i ≥ 0, and we prove that σ could be extended to be of length $i+2$. Since σ is infinite, there is a tableau rule R that appear at position i in σ. The goal position of this rule has the form EΦ. The proof is thus proceeds by an analysis of R.

- $R = R6$. In this case we have $\phi \in \varphi_i$ with φ_i is a positive (negative) node. φ_i has one negative (positive) node φ_j with $\neg\phi \in \varphi_j$. σ can be extended by adding φ_j.
- $R = R7$. Here we have $\phi = Perform(\alpha)\phi_1$ for some ϕ_1 and some action α. The node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $\phi_1 \in \varphi_j$, $s_{\varphi i} \rightarrow^{\alpha} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. According to the semantics of the *perform* operator and since $\Pi_{T'}(s_{\varphi i}) \vDash_T \neg\phi$ it follows that $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg\phi_1$. Thus, we can choose $\sigma(i+1) = \varphi_j$. It is clear that $\sigma_{T'}$ is a prefix of $\Pi_{T'}$.
- $R = R8$. This case is similar to the last case ($R = R7$) by substituting ϕ_1 by $SC(Ag_1, Ag_2, t, \varphi)$.
- $R = R9$. In this case we have $\phi = Create(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$. The current node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $SC(Ag_1, Ag_2, t, \varphi) \in \varphi_j$, $s_{\varphi i} \rightarrow^{C} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. It follows from the semantics of the *create* operator and from the fact that $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ that $\Pi_T(s_{0\varphi j}) \vDash_T \neg SC(Ag_1, Ag_2, \varphi)$. Thus, it is possible to extend σ such that 1 and 2 are always verified.
- $R = R10$. This case is similar to the last one by substituting the semantics of the *Create* operator by the semantics of the *Withdraw* operator.
- $R = R11$. Here we have $\phi = Satisfy(Ag_1, PC(Ag_1, Ag_2, t, \varphi))$. The current node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $\varphi \in \varphi_j$, $s_{\varphi i} \rightarrow^{Spc} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. It follows from the semantics of the *Satisfy* operator and from the fact that $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ that $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg\varphi$. In other words, this means that if an agent does not satisfy a propositional commitment, then the content of this commitment is false. Thus, it is possible to extend σ such that 1 and 2 are always verified.
- $R = R12$. This case is similar to the previous case. If an agent does not satisfy an action commitment about α, then $Perform(\alpha)p$ is not satisfied in the path $\Pi_{T'}(s_{0\varphi j})$.
- $R = R13$. Here we have $\phi = Violate(Ag_1, PC(Ag_1, Ag_2, t, \varphi))$. The current node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $\neg\varphi \in \varphi_j$, $s_{\varphi i} \rightarrow^{Vpc} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. It follows from the semantics of the *Violate* operator and from the fact that $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ that $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg(\neg\varphi)$. In other words, this means that if an agent does not violate a propositional commitment, then the content of this commitment is true. Thus, it is possible to extend σ such that 1 and 2 are always verified.
- $R = R14$. This case is similar to the case of $R13$.
- $R = R15$. In this case we have $\phi = Reactivate(Ag_1, SC(Ag_1, Ag_2, t, \varphi))$. The current node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $SC(Ag_1, Ag_2, t, \varphi) \in \varphi_j$, $s_{\varphi i} \rightarrow^{R} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. It follows from the semantics of the *Reactivate* operator and from the fact that $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ that

$\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg(SC(Ag_1, Ag_2, t, \varphi))$. In other words, this means that if an agent does not reactivate a SC in a model, then this commitment is not satisfied in this model. Thus, it is possible to extend σ such that 1 and 2 are always verified.

- $R = R16$. This rule deals with the challenge action. Thus we have $PC(Ag_2, Ag_1, t, ?\varphi) \in \varphi_j$ with φ_j is the only successor of φ_i. Since the fact that an agent does not challenge a SC implies that this agent does not commit about $?\varphi$, it follows that $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg(PC(Ag_2, Ag_1, t, ?\varphi))$. Therefore, we can choose $\sigma(i+1) = \varphi_j$.
- $R = Rx_{17 \leq x \leq 18}$. In this case we can choose $\sigma(i+1) = \varphi_j$ with φ_j is the successor of φ_i in r and $SC(Ag_2, Ag_1, \varphi') \in \varphi_j$ ($\varphi' \in \{\varphi, \neg\varphi\}$). In this case we have $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg(SC(Ag_2, Ag_1, t, \varphi'))$. The informal explanation is as follows: if an agent does not accept (respectively refuse) the content φ of a SC, this agent does not commit about φ (respectively $\neg\varphi$).
- $R = Rx_{19 \leq x \leq 21}$. These cases are similar. We deal with only the justification one. For this action we have $\phi = Justify\text{-}content(Ag_1, PC(Ag_1, Ag_2, t, \varphi), \varphi')$. The current node φ_i has one successor φ_j labeled by $<q', s_{0\varphi j}>$ such that $PC(Ag_1, Ag_2, t, \varphi' \therefore \varphi) \in \varphi_j$, $s_{\varphi i} \rightarrow^{Jus} s_{\varphi j}$ and $s_{0\varphi j}$ is the first sub-state of $s_{\varphi j}$. It follows from the semantics of the *Justify-content* operator and from the fact that $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ that $\Pi_{T'}(s_{0\varphi j}) \vDash_T \neg(PC(Ag_1, Ag_2, t, \varphi' \therefore \varphi))$. In other words, this means that if an agent does not justify a SC in a model, then this agent does not commit about $\varphi' \therefore \varphi$. Thus, it is possible to extend σ by φ_j such that 1 and 2 are always verified.
- $R = Rx_{22 \leq x \leq 23}$. These two cases are straightforward using the semantics of PC and AC.
- $R = R24$. this case is similar to the case of $R25$.
- $R = R25$. In this case we have $\phi = \phi_1 \wedge \phi_2$ for some $\phi \in \varphi_i$. Therefore, φ_i has two successors in r φ_j and φ_k with $\phi_1 \in \varphi_j$ and $\phi_2 \in \varphi_k$. Since $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ it follows that $\Pi_T(s_{\varphi j}) \vDash_T \neg\phi_1$ or $\Pi_T(s_{\varphi j}) \vDash_T \neg\phi_2$. Thus, σ can be extended by φ_j or by φ_k. It is clear that 1 and 2 are maintained.
- $R = R26$. In this case we have $\phi = \phi_1 \vee \phi_2$ for some $\phi \in \varphi_i$. Therefore, φ_i has one successor φ_j *or* φ_k with $\phi_1 \in \varphi_j$ and $\phi_2 \in \varphi_k$. Since $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ it follows that $\Pi_T(s_{\varphi j}) \vDash_T \neg\phi_1$ and $\Pi_T(s_{\varphi j}) \vDash_T \neg\phi_2$. Thus, we can extend σ by adding φ_j or φ_k. Constraints 1 and 2 are maintained.
- $R = R27$. In this case we have $\phi = ?\phi_1$ for some $\phi \in \varphi_i$. Therefore, φ_i has one successor φ_j with $\phi_1 \in \varphi_j$. Since $\Pi_{T'}(s_{\varphi i}) \vDash_T \neg\phi$ it follows that $\Pi_{T'}(s_{\varphi j}) \vDash_T \neg\phi_1$. Thus, we can extend σ by adding φ_j. Constraints 1 and 2 are maintained.
- $R = R28$. Here we have $\phi = \{X^-\phi_1, \ldots, X^-\phi_n)$ for some $\phi_1, \ldots, \phi_n$. φ_i labeled by $(q, s_{\varphi i})$ has one successor φ_j in r labeled by $<q', s_{\varphi j}>$ such that $q \rightarrow q'$ and $s_{\varphi j} \rightarrow s_{\varphi i}$ (notice that X^- is a past operator). Since $\Pi_T(s_{\varphi i}) \vDash_T \neg X^-\phi_k$ for $1 \leq k \leq n$, it follows that $\Pi_T(s_{\varphi j}) \vDash_T \phi_k$. Thus, $\sigma(i+1) = \varphi_j$.
- $R = R29$. This rule is applied when $\phi = \{X^+\phi_1, \ldots, X^+\phi_n)$ for some $\phi_1, \ldots, \phi_n$. φ_i labeled by $(q, s_{\varphi i})$ has one successor φ_j in r labeled by $<q', s_{\varphi j}>$ such that $s_{\varphi i} \rightarrow s_{\varphi j}$. Since $\Pi_T(s_{\varphi i}) \vDash_T \neg X^+\phi_k$ for $1 \leq k \leq n$, it follows that $\Pi_T(s_{\varphi j}) \vDash_T \neg\phi_k$. Thus, $\sigma(i+1) = \varphi_j$.

- $R = R30$. Here there is a $\phi \in \varphi_i$ such that $\phi = \phi_1 \therefore \phi_2$ for some $\phi \in \varphi_i$. Therefore, φ_i has one successor φ_j with $\phi_1 \wedge X^+(\neg\phi_1 \vee \phi_2) \in \varphi_j$. Since $\Pi_{T'}(s'_{\varphi i}) \vDash_T \neg\phi$ it follows from the semantics of $\therefore$ that $\Pi_{T'}(s'_{\varphi j}) \vDash_T \neg(\phi_1 \wedge X^+(\neg\phi_1 \vee \phi_2))$. We choose $\sigma(i+1) = \varphi_j$. It is clear that $\sigma_{T'}$ is a prefix of $\Pi_{T'}$.
- $R = R31$. In this case we have $\phi = \phi_1\ U^-\ \phi_2$. The node φ_i has one successor in r: φ_j ($\phi_2 \in \varphi_j$) or φ_k ($\{\phi_1, X^-\phi\} \in \varphi_k$). According to the semantics of U^- and since $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ it follows that either $\Pi_T(s_{\varphi j}) \vDash_T \neg(\phi_1 \vee \phi_2)$ or $\Pi_T(s_{\varphi k}) \vDash_T \phi_1 \wedge\neg\phi_2$ but $\Pi_T(s_{\varphi k}) \vDash_T \neg X^-\phi$. In the two cases, σ can be extended such that 1 and 2 are maintained.
- $R = R32$. This rule is used when $\phi = \phi_1\ U^+\ \phi_2$. The node φ_i has one successor in r: φ_j ($\phi_2 \in \varphi_j$) or φ_k ($\{\phi_1, X^+\phi\} \in \varphi_k$). According to the semantics of U^+ and since $\Pi_T(s_{\varphi i}) \vDash_T \neg\phi$ it follows that either $\Pi_T(s_{\varphi j}) \vDash_T \neg(\phi_1 \vee \phi_2)$ or $\Pi_T(s_{\varphi k}) \vDash_T \phi_1 \wedge\neg\phi_2$ but $\Pi_T(s_{\varphi k}) \vDash_T \neg X^+\phi$. In the two cases, σ can be extended such that 1 and 2 are maintained.

The last point in the proof of this lemma is to show that the path σ is unsuccessful. Let $i \geq 0$ be such that $\phi_1\ U^+\ \phi_2 \in \sigma(i)$ for some ϕ_1 and ϕ_2. According to Definition 8.17, we must show that is no $j \geq i$ such that $\phi_2 \in \sigma(j)$.

$\phi_1\ U^+\ \phi_2 \in \varphi_i$ $\qquad\qquad\qquad\qquad$ $\varphi_i = \sigma(i)$

$\Rightarrow$ (The constraint 2 is verified by the way σ is constructed)

$\quad \Pi_T(s_{\varphi i}) \vDash_T \neg(\phi_1\ U^+\ \phi_2)$

$\Rightarrow \Pi_T(s_{\varphi i}) \vDash_T \neg\phi_1 \wedge \neg\phi_2$ or $\Pi_T(s_{\varphi i}) \vDash_T \phi_1 \wedge \neg\phi_2 \wedge \neg X^+(\phi_1\ U^+\ \phi_2)$

$\Rightarrow \forall i \geq j\ \Pi_T(s_{\varphi j}) \vDash_T \neg\phi_2$

$\Rightarrow \phi_2 \in \varphi_j$.

□

Now, we prove the third element of the correctness theorem that deals with the acceptance condition.

Lemma 8.21 *An infinite path σ in a run r of B_ψ is successful iff it satisfies the generalized Büchi condition.*

Proof

The proof follows from the definition of $F_{\phi 1\ U+\ \phi 2}$.

1) "⇒" Assume that σ is successful. Suppose that $\phi_1\ U^+\ \phi_2 \in p_i$. Since the path is successful, there exists a p_j, $j \geq i$ such that $\phi_2 \in p_j$. Hence, for any i we can find a $j \geq i$ such that $p_j \in F_{\phi 1\ U+\ \phi 2}$. It follows that φ_j is an accepting state that occurs infinitely often.

2) "⇐" Assume that σ satisfies the generalized Büchi condition. Suppose that $\phi_1\ U^+\ \phi_2 \in p_i$ for some i. Since the path satisfies the Büchi condition, there exists a $j \geq i$ such that $p_j \in F_{\phi 1\ U+\ \phi 2}$. Two cases can be distinguished:

a. if $\phi_2 \in p_j$, then σ is successful.

b. if $\phi_2 \notin p_j$, then according to the semantics of U^+ and the rule $R32$, there exists a $i \leq k \leq j$ such that $\phi_2 \in p_k$. Therefore, σ is successful.

□

8.5.5 Examples

In this section we illustrate the construction of an ABTA for two formulae. The first formula is a propositional one. The second formula is an action one.

a. Example 1

Let us consider the following propositional formula: $E(G^+ F^+ p)$. The tableau of this formula is illustrated by Figure 8.6. The first rule we can apply is *R*32 labeled by "$\vee$" for the until formula (G^+ is an abbreviation defined from U^+). The second rule is also *R*32 for F^+p (F^+ is also an abbreviation defined from U^+). Thereafter rules *R*24 and *R*29 can be applied, etc.

The ABTA obtained from this tableau is illustrated in Figure 8.7. In this ABTA, states (1), (3), (5) and (6) are the acceptance states according to Definition 8.6. The formula ϕ we consider is the following: $\phi = True\ U^+\ p \equiv F^+p$. Notice that ϕ and $X^+\phi$ do not appear in these states. State (5) is the acceptance state in the finite case. On the other hand, ϕ appears in states (2) and (7), and $X^+\phi$ appears in state (4). Therefore, these states are not in F_ϕ. The path Π = (1, (2, 4, 7)*) is not a valid proof of $E(G^+F^+p)$. However, a path that visits infinitely often the states (1), (3) and (6) is a valid (infinite) proof. The reason is that in such a path there is always a chance to meet the proposition *p* (state (3)). Therefore, this path satisfies the Büchi condition. The Büchi condition is not satisfied in the path Π since there is no chance to visit infinitely often a state containing *p*.

$\vee : E(G^+ F^+ p)$ (1)

$\vee : E(F^+ p, X^+ G^+ F^+ p)$ (2)

$<\equiv>: E(p, X^+ G^+ F^+ p)$ (3) | $X^+ : E(X^+ F^+ p, X^+ G^+ F^+ p)$ (4)

p (5) | $X^+ : E(X^+ G^+ F^+ p)$ (6) | $\vee : E(F^+ p, G^+ F^+ p)$ (7)

$E(G^+ F^+ p)$ | $E(F^+ p, X^+ G^+ F^+ p)$

Figure 8.6. The tableau for $E(G^+F^+p)$

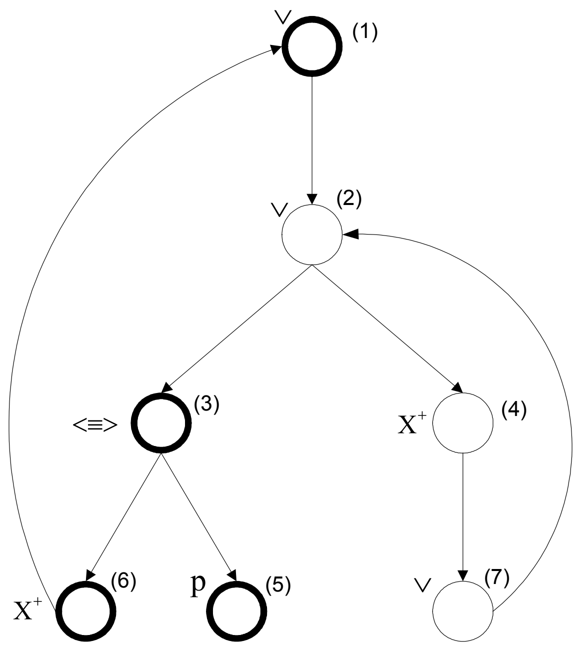


Figure 8.7. The ABTA of the formula $E(G^+F^+p)$

b. Example 2

In this section we consider the following action formula from $DCTL^*_{CAN}$:

$$AG^+\mathbf{(}\textit{Challenge-content}(Ag_2, PC(Ag_1, Ag_2, t, \phi)) \Rightarrow F^+\textit{Justify-content}(Ag_1, PC(Ag_1, Ag_2, t, \phi), \phi')\mathbf{)}.$$

In order to simplify this formula, we use *Ch* for *Challenge-content* and *Jus* for *Justify-content*. The tableau of this formula is illustrated by Figure 8.8. The associated ABTA of this formula is given by Figure 8.9. This formula is equivalent to the formula:

$$AG^+\mathbf{(}\neg Ch(Ag_2, PC(Ag_1, Ag_2, t, \phi)) \vee F^+Jus(Ag_1, PC(Ag_1, Ag_2, t, \phi), \phi')\mathbf{)}.$$

$$\neg : AG^{+}(\neg Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \vee F^{+}Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi')) \quad (1)$$

$$\vee : EF^{+}(Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (2)$$

$$<Ch>: E(Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (3)$$

$$<X^{+}>: EX^{+}(F^{+}(Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi')))) \quad (4)$$

$$[PC_{Ag2}] : E(PC(Ag_{2}, Ag_{1}, t, ?\phi) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (5)$$

$$EF^{+}(Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (2)$$

$$? : E((?\phi) \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (6)$$

$$<\equiv>: E(\phi \wedge G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (7)$$

$$\varphi \; (8) \qquad \vee : E(G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (9)$$

$$<\neg Jus>: E(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'), X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (10)$$

$$[PC_{Ag1}] : E(PC(Ag_{1}, Ag_{2}, t, \phi' \therefore \phi), X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (11)$$

$$\wedge : E(\phi' \therefore \phi, X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (12)$$

$$<\equiv>: E(\phi', X^{+}(\neg\phi' \vee \phi), X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (13)$$

$$\varphi' \; (14) \qquad X^{+} : E(X^{+}(\neg\phi' \vee \phi), X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (15)$$

$$<\equiv>: E((\neg\phi' \vee \phi), X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (16)$$

$$\neg\varphi' \vee \varphi \; (17) \qquad X^{+} : E(X^{+}G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (18)$$

$$E(G^{+}(\neg Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))) \quad (9)$$

Figure 8.8. The tableau for $AG^{+}(Ch(Ag_{2}, PC(Ag_{1}, Ag_{2}, t, \phi)) \Rightarrow F^{+}Jus(Ag_{1}, PC(Ag_{1}, Ag_{2}, t, \phi), \phi'))$

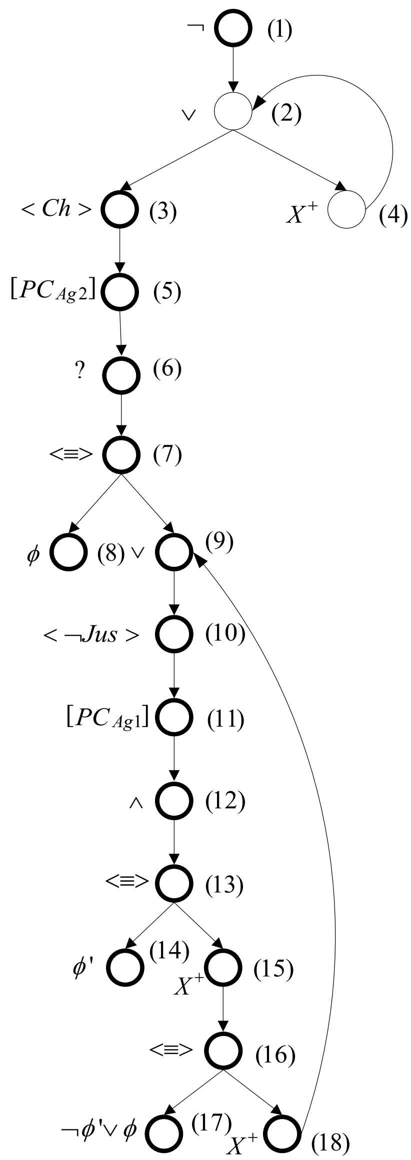


Figure 8.9. The ABTA for The formula
$AG^{+}(Ch(Ag_2, PC(Ag_1, Ag_2, t, \phi)) \Rightarrow F^{+}Jus(Ag_1, PC(Ag_1, Ag_2, \phi), \phi'))$

The first rule we can apply is *R*6 labeled by "¬". We obtain then the formula (2) of Figure 8.8. From this formula we obtain the formula ϕ that we consider in order to compute the acceptance states:

$\Phi = F^{+}(Ch(Ag_2, PC(Ag_1, Ag_2, t, \phi)) \wedge G^{+}(\neg Jus(Ag_1, PC(Ag_1, Ag_2, t, \phi), \phi')))$.

In the ABTA of Figure 8.9 state (1) and states from (3) to (18) are the acceptance states according to Definition 8.6. States (2) and (4) are not acceptance states. Because only the first state is labeled by¬, all finite and infinite paths are negative paths. Consequently, the only infinite path that is a valid proof of the formula Φ is (1, (2, 4)*). In this path there is no acceptance state that occurs infinitely often. Therefore, this path satisfies the Büchi condition. The path visiting the state (3) and infinitely often the state (9) does not satisfy the formula because there is a challenge action (state (3)), and globally no justification action of the content of the challenged propositional commitment (state (9)).

8.5.6 Model Checking Algorithm

Our model checking algorithm for verifying that a dialogue game protocol satisfies a given property and checks that it respects the semantics of the underlying communicative acts is based on the procedure proposed by (Bhat and Cleaveland, 1996). Like the algorithm proposed by (Courcoubetis et al., 1992), our algorithm explores the product graph of an ABTA for DCLT*$_{CAN}$ and a transition system for a dialogue game. This algorithm is on-the-fly (or local) algorithm that consists of checking if a transition system is accepted by an ABTA. This ABTA model checking is reduced to the emptiness of the Büchi automata (Vardi and Wolper, 1986).

Let $T = <S, Lab, \wp, L, Act, \rightarrow, s_0>$ be a transition system for a dialogue game and let $B = <Q, l, \rightarrow, q_0, F>$ be an ABTA for DCTL*$_{CAN}$. The procedure consists of building the ABTA product $B_\otimes$ of T and B while checking if there is a successful run in $B_\otimes$. The existence of such a run means that the language of $B_\otimes$ is non-empty. The automaton $B_\otimes$ is defined as follows: $B_\otimes = <Q \times S, \rightarrow_{B\otimes}, q_{0B\otimes}, F_{B\otimes}>$. There is a transition between two nodes $<q, s>$ and $<q', s'>$ iff there is a transition between these two nodes in some run of B on T. Intuitively, $B_\otimes$ simulates all the runs of the ABTA. The set of accepting states $F_{B\otimes}$ is defined as follows: $q_{0B\otimes} \in F_{B\otimes}$ iff $q \in F$.

Unlike the algorithms proposed in (Courcoubetis et al., 1992) and (Bhat and Cleaveland, 1996), our algorithm uses only one depth-first search (DFS) instead of two. This is due to the fact that our algorithm explores directly the product graph using the sign of the nodes (positive, negative or neutral). In addition, unlike the algorithm proposed in (Bhat and Cleaveland, 1996), our algorithm does not distinguish between recursive and non-recursive nodes. Therefore, we do not take into account the strongly-connected components in the ABTA, but we use a marking algorithm that works on the product graph.

The pseudo-code of this algorithm is given in Figure 8.10. The idea is to construct the product graph while exploring it. However, in order to make it easy to understand, we omit the instructions relative to the addition of nodes in the product graph. The construction procedure is directly obtained from Definition 8.4. The algorithm uses the label of nodes in the ABTA, and the transitions in the product graph obtained from the transition system and the ABTA as explained in Definition 8.4.

```
DFS(v = (q, s)): boolean {
    if v marked visited {
      if (sign(v) =  "+" and not accepting(v)) or (sign(v) = "-" and accepting(v))
        return false
    } // end of if v marked visited
    else {
      mark v visited
      switch(l(q)) {
          case (p ∈ Φp):
              switch(sign(v)) {
                  case("+"): if s is a sub-state and l(q) ∉L'(s) return false
                  case("-"): if s is a sub-state and ¬l(q)) ∉L'(s) return false
                  case("neutral"): return false
              } // end of switch(sign(v))
          case(∧):
              if s is a leaf return false
              else
                  switch(sign(v)) {
                      case(neutral): for all v'' ∈ {v' / v →B⊗ v'} if not DFS(v'') return false
                      case("+"): for all v'' ∈ {v' / v →B⊗ v'} if not DFS(v'') return false
                      case("-"): for all v'' ∈ {v' / v →B⊗ v'} if DFS(v'') return true
                                 return false
                  } // end of switch(sign (v))
          case(∨):
              if s is a leaf return false
              else
                  switch(sign(v)) {
                      case(neutral): for all v'' ∈ {v' / v →B⊗ v'} if DFS(v'') return true
                                     return false
                      case("+"): for all v'' ∈ {v' / v →B⊗ v'} if DFS(v'') return true
                                 return false
                      case("-"): for all v'' ∈ {v' / v →B⊗ v'} if not DFS(v'') return false
                  } // end of switch(sign (v))
          case(<•>):
              if s is a leaf return true
              else for the v'' ∈ {v' / v →B⊗ v'} if not DFS(v'') return false
          case(X+, PC_Ag, AC_Ag, <≡>, ?):
              if s is a leaf return false
              else for the v'' ∈ {v' / v →B⊗ v'} if not DFS(v'') return false
      } // end of switch(l(q))
    } // end of else
    return true }
```

Figure 8. 10. Exploring product graph algorithm

In order to decide if the ABTA contains an infinite successful run, all the explored nodes are marked "visited". Thus, when the algorithm explores a visited node, it returns false if the infinite path is not successful. If the node is not already visited, the algorithm tests if it is a leaf. In this case, it returns false if the node is a non-successful leaf. If the explored node is not a leaf, the algorithm calls recursively the function DFS in order to explore the successors of this node. If this node is labeled by "$\wedge$", and signed neutrally or positively, then DFS returns false if one of the successors is false. However, if the node is signed negatively, DFS returns false if all the successors are false. A dual treatment is applied when the node is labeled by "$\vee$". We note that if the DFS does not explore a false node (i.e. it does not return false), then it returns true.

Theorem 8.22 (correctness) *Let B an ABTA and T a transition system. DFS*(q_0, s_0) *returns true if and only if T is accepted by B.*

Proof
This theorem follows from Theorem 8.15 and Definition 8.5. Indeed, DFS returns true if and only if all the leaves are successful, and all the infinite paths are successful. The reason is that DFS returns true if and only if it does not find any unsuccessful leaf and any unsuccessful infinite path.
□

Figure 8.11 illustrates the automaton $B_\otimes$ resulting from the product of the transition system of Figure 8.5 (TS[8.5]) and the ABTA of Figure 8.9 (ABTA[8.9]). In order to check if the language of this automaton is empty, we check if there is a successful run. The idea is to verify if $B_\otimes$ contains an infinite path visiting the state (3) and infinitely often the state (9) of ABTA[8.9]. If such a path exists, then we conclude that the formula is not satisfied by TS[8.5]. Indeed, the only infinite path of $B_\otimes$ is successful because it does not touch any accepted state and all leaves are also successful. For instance, the leaf labeled by (<*Ch*>,s_0) is successful since there is no state s_i such that $s_0 \rightarrow^{Ch} s_i$. The leaf labeled by ($\neg\phi' \vee \phi$, $s_{3,4}$) is successful because it is a positive leaf and $s_{3,4} \models \neg\phi' \vee \phi$. Therefore, TS[8.5] is accepted by ABTA[8.9]. Consequently, TS[8.5] satisfies the formula and respects the semantics of challenge and justification actions.

We conclude this section by discussing the worst-case time complexity of our model checking technique.

Lemma 8.23 *Let ψ be a DCTL*$_{CAN}$ formula, ant let* B_ψ = <Q, l, $\rightarrow$, q_0, F > be *the ABTA obtained by the translation procedure. Then* $|B_\psi| < 2^{|\psi|}$.

Proof
From the transition procedure, each formula ψ' in the tableau is a sub-formula of ψ. The formula ψ is decomposed into a set of sub-formulae using the tableau rules. The nodes in the ABTA are labeled by the operators from the sub-formulae and there is a transition from a node φ to a node φ' if the formula corresponding to φ' is a sub-formula of the one

corresponding to φ. Since for every sub-formula ψ' of ψ we have $\psi' \subseteq CL(\psi)$ and $|CL(\psi)| < |\psi|$ (from Lemma 8.11), it follows that $|B_\psi| < 2^{|\psi|}$.
□

The complexity of the transition procedure is thus exponential in the size of the formula ($O(2^{|\psi|})$). However, if ψ is a $DCTL_{CAN}$ formula, $|B_\psi|$ is bounded by $|\psi|$. The complexity is then linear in the size of the formula. This result follows from the fact that in $DCTL_{CAN}$ we have only state formulae.

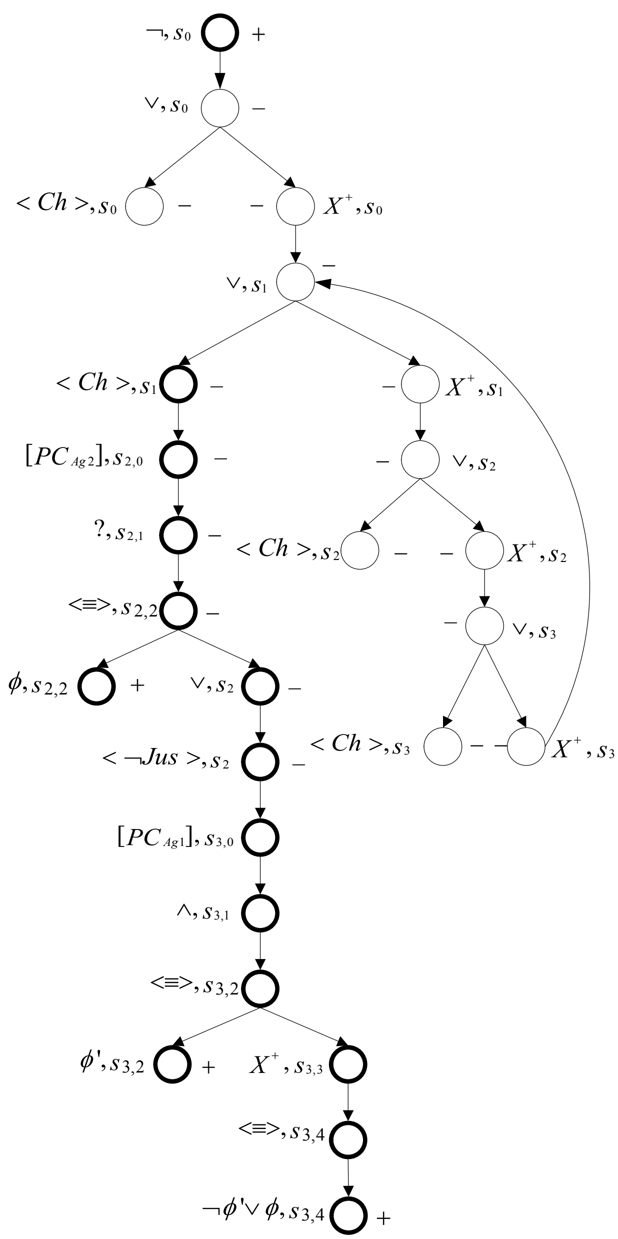


Figure 8.11. The ABTA product of the TS of Figure 8.5 and the ABTA of Figure 8.9

Lemma 8.24 *Let T = <S, Lab, $\wp$, L, Act, $\rightarrow$, s_0> be a transition system for a dialogue game, and let B_ψ = <Q, l, $\rightarrow$, q_0, F > be an ABTA for ψ. The time complexity of the model checking algorithm is bounded by* $|T| \times |B_\psi|$ where $|T| = |S| + |\wp| + |\rightarrow|$ and $|\wp|$ *is the number of sub-states in all semantic transition systems of T.*

Proof

The algorithm is based on a product graph of the ABTA B_ψ and the transition system T. The size of this product is bounded by $|T| \times |B_\psi|$. Like the algorithms proposed in (Courcoubetis et al., 1992) and (Bhat and Cleaveland, 2001), our algorithm marks nodes and determines if an accepting state is reachable from itself. This algorithm visits each state once and there are $|S| \times |Q|$ recursive calls to a depth-first search algorithm. We note also that the ABTA we use is an *and-restricted* one. In an and-restricted ABTA only one of the children of a node labeled by $\wedge$ can have his truth values determined by recursive calls to search algorithm (Bhat and Cleaveland, 2001). The run time of the algorithm is thus proportional to the size of the product graph, i.e. $O(|T| \times |B_\psi|)$.

□

The worst-case time complexity of our model-checking technique is therefore linear in the size of the model and exponential in the size of the formula to be checked.

8.6 Related Work

The verification problem has recently begun to find a significant audience in the MAS community. Rao and Georgeff (1993) defined three variants of propositional BDI (beliefs, desires and intention) logics for MAS and they proposed basic model checking algorithms for these logics. These algorithms are an adaptation of the algorithms for CTL and CTL*. van der Hoek and Wooldridge (2002) proposed some techniques for model checking temporal epistemic *properties of MAS using an epistemic logic (logic of knowledge). They proposed a technique in order to reduce* the model checking of this logic to the model checking of linear temporal logic (LTL). Benerecetti and Cimatti (2002) proposed a general approach for model-checking MAS based on CTL together with modalities for BDI attitudes. Wooldridge and his colleagues (2002) presented the MABLE language for the specification and the verification of MAS. Agents specified in this language have data structures corresponding to BDI. MABLE is automatically translated into Promela, the language of SPIN model checker of LTL (Holzman, 1997). Bordini et al. (2003a) addressed the verification problem of MAS specified using AgentSpeak (Rao, 1996). They used a finite state version of this language and they showed how programs written in it can be automatically transformed into Promela. In order to specify the properties to be verified, the authors used a simplified form of BDI Logic. These specifications are then translated to LTL formulae. Propositional attitudes are modeled as Promela data structures. Bordini et al. (2003b) proposed another alternative for model checking AgentSpeak by translating this language to Java in order to apply JPF2, a Java model checker (Visser et al., 2000). Penczek and Lomuscio (2003) proposed a framework for verifying temporal and epistemic properties of MAS. They proposed a bounded model checking algorithm for branching time logic for knowledge (CTLK). The basic idea of bounded model checking is to search for a counterexample in executions whose length is bounded by some integer *k*. If no bug is found then one increases *k* until either a bug is found, the problem becomes intractable, or some pre-known upper bound is reached (this bound is called the *Completeness Threshold* of the design. In a similar way, Raimondi and Lomuscio (2004) implemented an algorithm to verify epistemic CTL properties of MAS via ordered binary decision diagrams (Clarke et al., 1999). Kacprzak and her colleagues (2004b) also investigated the problem of verifying epistemic properties using CTLK by means of an unbounded model checking algorithm based on the technique proposed by McMillan (2002). Kacprzak and Penczek (2004a) addressed the problem of verification of game-like structures by means of unbounded model checking using alternating-time temporal logic (Alur et al., 1997). There are many differences between all these proposals and the work presented in this chapter that we can summarize as follows:

1- These proposals are based on BDI and epistemic logics that stress the agents' private mental states, whereas our work uses a logic highlighting the public states reflecting the agents' interactions expressed in terms of social commitments and argumentation relations.

2- Our model checking algorithm allows us to verify not only the system's temporal properties but also the action properties specified using dynamic logic.

3- The technique that we use is based on the tableau method and is different from the techniques used for LTL, CTL and CTL*.

Complementarily, the verification of agent communication protocols has been addressed by some research work. Endriss and his colleagues (2003) dealt with the problem of checking and possibly enforcing conformance to agent communication protocols. They proposed abductive logic-based agents and some means of determining whether or not these agents behave in conformance to the defined protocols. Baldoni et al. (2004) addressed the problem of verifying that a given protocol implementation conforms to its specification. They studied a special case in which protocols are implemented using a logical language and specified using AUML. These approaches are different from our proposal in the sense that they are not based on model checking techniques and they do not address the problem of verifying whether or not a protocol satisfies a given property.

In (Huget and Wooldridge, 2004), the problem of checking that agents correctly implement the semantics of an agent communication language is addressed. Huget and Wooldridge used a variation of the MABLE programming language to define a pre/post conditions semantics of ACL performatives and showed that the compliance to ACL semantics can be reduced to a conventional model checking problem. Walton (2004) applied model checking techniques in order to verify the correctness of protocol communication. The author defined a protocol language and used the SPIN model checker to verify LTL properties of this language. The model checking technique used by these two proposals are based on LTL whereas our technique is based on CTL* and dynamic logic. In addition, our approach is based on a new algorithm and not on the translation of the specification language to existing model checker language.

Recently, Giordano et al. (2004) addressed the problem of specifying and verifying agent interaction protocols using a Dynamic Linear Time Temporal Logic (DLTL) (Henriksen and Thiagarajan, 1999). These protocols are specified using temporal constraints representing permissions and SC. The authors addressed three kinds of verification problems: 1) the compliance of an execution history of a protocol to its specification, 2) the satisfaction of a property in the protocol, 3) the compliance of agents to the protocol. They showed that these problems can be solved by model checking in DLTL. This model checking technique uses a tableau-based algorithm for obtaining a Büchi automaton from a formula in DLTL (Giordano and Martelli, 2004). Although this work is close to our proposal, the model and the automata associated to the checked formulae used in the two techniques are different. Indeed, there are four main differences between these two approaches:

1- The protocols (the models) we dealt with are dialogue game protocols described as a combination of dialogue games (Bentahar et al, 2004a, 2004d) (see Chapter 9) and specified using actions that agents apply on SC. However, the protocols used in (Giordano et al., 2004) are abstract protocols specified in terms of the effects of communicative actions, some precondition laws, and some causal law.

2- The model checking technique proposed in (Giordano and Martelli, 2004) uses classical Büchi automaton that is constructed using a tableau-like procedure. This procedure is based on propositional rules and exploits two axioms defining the semantics of the indexing until operator. Our technique is different because it is based on ABTA and not on traditional Büchi automaton. In addition, the construction of this automaton uses proof rules that define the tableau semantics of the different formulae and not propositional rules.

3- Our approach is based not only on SC like (Giordano et al., 2004), but also on an argumentation theory. Consequently, our protocols are more suitable for autonomous agents. The reason is that agents can make decisions using their argumentation systems.

4- The dynamic part in our logic is reflected by an action theory, i.e. by the actions that agents perform. In our logic we deal with action formulae, whereas in DLTL, the dynamic part is represented by regular programs and by indexing the *until* operator with these programs.

8.7 Conclusion

In this chapter, we have addressed the verification problem of dialogue game protocols. We proposed a new model checking technique allowing us to verify both the correctness of the protocols and the agents' compliance to the semantics of the communicative acts. This technique uses a combination of an automata-based and a tableau-based algorithm to verify temporal and action specification. The formal properties to be verified are expressed in $DCTL^*_{CAN}$ logic and translated to ABTA using tableau rules. Our model checking algorithm that works on a product graph is an efficient on-the-fly procedure.

The semantics that we used in this chapter is a simplified version of the semantics defined in Chapter 7. This simplified semantics does not express the satisfaction of formulae in a given theoretical model, but expresses the decomposition of these formulae to sub-formulae. Consequently, what we can verify is the fact that a given protocol satisfies or not a given property and the fact that agents use the same decomposition of formulae. For example, in terms of the semantics of a social commitment, we can only verify if for the debtor there is a state in which the commitment content is true. This supposes that the content is a state formula, and not a path formula as defined in Chapter 7. Improving this simplified version of semantics is in our future work.

Chapter 9*

Application: Specifying and Implementing a Persuasion Dialogue Game Protocol

In this chapter, we present an application of our pragmatic approach. We propose a new persuasion dialogue game protocol for agent communication specified using this approach. We show how this protocol is modeled by the CAN framework. Our dialogue game protocol is specified by indicating its entry conditions, its dynamics and its exit conditions. In order to solve the problem of the acceptance of arguments, the protocol integrates the concept of agents' trustworthiness in its specification. The chapter proposes a set of algorithms for the implementation of the persuasion protocol and discusses their termination, complexity and correctness. The chapter addresses also the implementation issues of our protocol using logic programming and an agent-oriented platform.

9.1 Introduction

Research in agent communication protocols has received much attention during the last years. Protocols are means of achieving meaningful interactions. In multi-agent systems (MAS), agents use protocols to guide their interactions with each other. Protocols describe the allowed communicative acts that agents can perform when conversing. These protocols specify the rules governing a dialogue between agents in MAS.

Protocols for multi-agent interaction need to be flexible because of the open and dynamic nature of MAS. Traditionally, these protocols are specified as finite state machines or Petri nets without taking into account the agents' autonomy. Therefore, these protocols are not flexible enough to be used in open MAS (Maudet and Chaib-draa, 2002). To solve this problem, several researchers proposed protocols using dialogue games (Dastani et al., 2000) (Dignum et al., 2001) (Maudet and Chaib-draa, 2002) (McBurney and Parsons, 2002) (see Chapter 3 for more details). Dialogue games are interactions between players, in which each player moves by performing utterances according to a pre-defined set of roles (McBurney and Parsons, 2002). The flexibility is achieved by combining different games to construct complete and more complex protocols.

---

* We would like to thank John-Jules Ch. Meyer from Utrecht University, Intelligent Systems Group, Claude Bélisle from Laval University, Department of Mathematics and Statistics and Iyad Rahwan from the University of Melbourne for their interesting comments about the matter presented in this chapter. The computational model introduced in this chapter is published in (Bentahar et al., 2004a, 2004d).

In this chapter, we propose a persuasion protocol specified using a set of dialogue games. We formalize these dialogue games as a set of conversation policies. Conversation policies are declarative specifications that govern communication between autonomous agents (Greaves et al., 2000). Indeed, protocols specified using, for example, finite state machines are not flexible in the sense that agents must respect the whole protocol from the beginning to the end without reasoning about them. Thus, we propose to specify these protocols by small conversation policies that can be logically put together using a combination of dialogue games.

On the other hand, the protocols described in the literature are often specified by pre/post conditions. These protocols often neglect the decision-making process that allows agents to accept or to refuse an utterance. The protocols based on formal dialectics (Elvang-Goransson et al., 1993), (Prakken, 2001), (Amgoud et al., 2000a, 2000b) use the argumentation as a way of expressing decision-making. However, the sole use of argumentation does not make it possible to solve a decision-making problem well. We think that other social elements such as agents' trustworthiness must also be taken into account.

The contribution of this chapter is the proposition of a new approach for specifying protocols for agent communication. A new persuasion dialogue game protocol is specified and implemented following this approach. This protocol is modeled using our pragmatic approach based on commitments and arguments. It is flexible in the sense that it is specified by small conversation policies that can be combined and in the sense that agents can reason about this protocol using their argumentation systems and the trustworthiness notion. The algorithms implementing this protocol are specified using the CAN framework. This protocol is characterized by the fact that it integrates the agents' trustworthiness as a component of the decision-making process. Indeed, this chapter presents three main results:
1- A new formal language for specifying a persuasion dialogue game protocol as a combination of conversation policies.
2- A termination proof of the protocol based on the tableau method described in Chapter 8.
3- An implementation of the specification using an agent-oriented and logic programming framework.

The rest of this chapter is organized as follows. In Section 9.2, we address the specification of our persuasion protocol. We present the protocol form, the specification of each dialogue game and the protocol dynamics. We also present the different algorithms implementing these dialogue games, develop a termination proof, and discuss the correctness and complexity analysis. In Section 9.3, we highlight the importance of agents' trustworthiness and present our model of this trustworthiness. In Section 9.4, we describe some issues in the implementation of the trustworthiness model and dialogue games. In Sections 9.5, 9.6, and 9.7, we compare our protocol to related work, we discuss the flexibility of this protocol, and we conclude.

9.2 Specification of Dialogue Games

9.2.1 Philosophical Foundations

According to the classification proposed by Walton and Krabbe (1995), each type of dialogue has an initial situation and the goal of the dialogue is to change this situation in a particular way. Figure 9.1 illustrates the initial situation as well as the goal of the persuasion dialogue.

In the same context, Vanderveken (2001) proposed a *logic of discourse* in which there are only four possible discursive goals that speakers can attempt to achieve by conversing. These goals are: descriptive, deliberative, declaratory and expressive goals. Persuasion dialogue is a sub-type of the dialogue types having a descriptive goal. In his typology, Vanderveken argued that each dialogue type with a discursive goal has a mode of achievement of the discursive goal and preparatory conditions. The mode of achievement imposes a certain sequence of speech acts. For a persuasion dialogue, a certain sequence of defense utterances, questions and answers is needed for the successful implementation of such a dialogue. Preparatory conditions determine a structured set of presuppositions related to the discursive goal. The persuasion dialogue has the preparatory conditions that there is a conflict between the agents' points of view and that each agent has the capacity to defend its point of view.

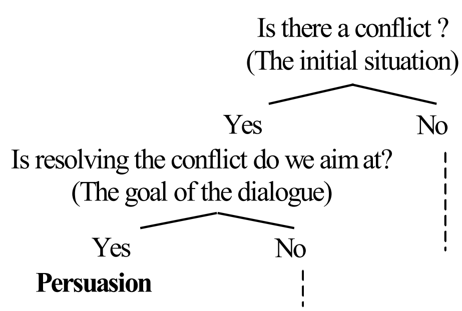


Figure 9.1. Goal and initial situation of the persuasion dialogue

In addition, in the domain of artificial intelligence and law, many computational and logical models of argument and debate, and of reasoning with conflicting information have been proposed (Prakken, 1997), (Prakken and Sartor, 1998), (Bench-Capon et al., 2003). Prakken and Sartor (1998) introduced a dialectical proof theory for an argumentation framework. A proof of a formula takes the form of a dialogue tree, in which each branch of the tree is a dialogue and the root of the tree is an argument for the formula. The idea is that every move in a dialogue consists of an argument based on the input theory, where each stated argument attacks the last move of the opponent in a way that meets the player's burden of proof.

Our persuasion protocol is defined by specifying its entry conditions, its exit conditions, and its dynamics. Entry conditions correspond to the initial situation of the dialogue and to the preparatory conditions. Exit conditions correspond to the final situation that makes it possible to determine if the dialogue goal is achieved or not. The dynamics specifies the different types of actions that can be performed by agents so that each agent can achieve its goal. The dynamics correspond to the mode of achievement of the discursive goal. It also corresponds to the dialectical proof theory where the root is the persuasion subject. The dynamics is specified by a set of initiative / reactive dialogue games. An initiative game involves creating a new commitment. A reactive game consists in taking position on an existing commitment (acceptance, refusal, challenge, defense, etc.).

9.2.2 CAN and the Persuasion Protocol

A Entry Conditions

As illustrated by Figure 9.1, the entry condition of the persuasion protocol is a conflict of point of view. This is translated in the CAN formalism by the creation of a commitment $SC(Id_x, Ag_1, Ag_2, p)$ by an agent Ag_1 and the refusal of this commitment by an agent Ag_2. Formally, the initial situation is reflected as follows:

$F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_x, p)) = \{(Create, t_i)\}$
$F_\Omega(Ag_2\ SC(Ag_1, Ag_2, t_x, p)) = \{(Refuse\text{-}content, t_{i+1})\}$
$F_\Omega(Ag_2, SC(Ag_2, Ag_1, t_{x+1}, \neg p)) = \{(Create, t_{i+1})\}$

B Dynamics

Generally, the persuasion dialogue takes the form of a sequence of attacks and defenses where each agent tries to defend its point of view or attack the point of view of its partner. This dialogue can also contain questions and answers (dialogue game of information seeking). In the CAN formalism, this results in the creation of commitments that defend or attack the initial commitment and other commitments and argumentation relations. The dialogue games of information seeking can be represented by challenge actions and argumentation relations. Formally, the dialogue dynamics can be expressed by a combination of the following functions:

$F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_y, q)) = \{(Create, t_j)\}$
$F_{E\Sigma}(SC(Ag_1, Ag_2, t_y, q), SC(Ag_1, Ag_2, t_x, p)) = (Defend\text{-}content, t_j)$
$F_\Omega(Ag_2, SC(Ag_2, Ag_1, t_z, r)) = \{(Create, t_k)\}$
$F_{E\Sigma}(SC(Ag_2, Ag_1, t_z, r), SC(Ag_1, Ag_2, t_x, p)) = (Attack\text{-}content, t_k)$

where p, q, r are propositional formulae.

Information seeking can be, for example, represented by:

$F_\Omega(Ag_2, SC(Ag_1, Ag_2, t_y, q)) = \{(Challenge\text{-}content, t_l)\}$
$F_\Omega(Ag_1, SC(Ag_1, Ag_2, t_z, r)) = \{(Create, t_{l+1})\}$
$F_{E\Sigma}(SC(Ag_1, Ag_2, t_z, r), SC(Ag_1, Ag_2, t_y, q)) = (Justify\text{-}content, t_{l+1})$

C Exit Conditions

The persuasion dialogue terminates either if the conflict is resolved, or with a situation in which each agent does not accept the argument of the other. In this case the protocol terminates with an unresolved conflict. The conflict is resolved when one of the two agents adopts the point of view of its partner. In the CAN formalism, this results in the acceptance of the initial commitment $SC(Ag_1, Ag_2, t_x, p)$ (respectively $SC(Ag_2, Ag_1, t_{x+1}, \neg p)$) by Ag_2 (respectively Ag_1). This implies the cancellation of all commitments attacked $SC(Ag_1, Ag_2, t_x, p)$ (respectively $SC(Ag_2, Ag_1, t_{x+1}, \neg p)$). Formally, if Ag_2 accepts $SC(Id_x, Ag_1, Ag_2, p)$, the final situation is described as follows:

$(Accept\text{-}content, t_m) \in F_{\Omega}(Ag_2, SC(Ag_1, Ag_2, t_x, p)) \wedge$
$(\forall t_y, q, t_l\text{: } F_{E\Sigma}(SC(Ag_2, Ag_1, t_y, q), SC(Ag_1, Ag_2, t_x, p)) = (Attack\text{-}content, t_l) \Rightarrow$
$(Withdraw, t_m) \in F_{\Omega}(Ag_2, SC(Ag_2, Ag_1, t_y, q)))$

Agents must also update their knowledge bases by removing the attacked and non-defended arguments and adding the new accepted arguments. When the two agents mutually refuse the argument of the other, the protocol stops because the conflict cannot be resolved.

9.2.3 Protocol Form

Our persuasion protocol is specified as a set of initiative / reactive dialogue games that are specified as a combination of conversation policies. In accordance with our pragmatic approach (see Chapters 5 and 6), the game moves are considered as actions that agents apply to commitments, to their contents and to arguments. A conversation policy is specified as follows:

$$Action\_Ag_1 \xrightarrow{Cond} Action\_Ag_2$$

This specification indicates that if an agent Ag_1 performs the action $Action\_Ag_1$, and that the condition *Cond* is satisfied, then the interlocutor Ag_2 will perform the action $Action\_Ag_2$ afterwords. The condition *Cond* is expressed in terms of the possibility of generating an argument from the agent's argumentation system and in terms of the interlocutor's trustworthiness.

Before introducing some formal notation we use in our specification, we notice that we distinguish between arguments that an agent has (private arguments) and arguments that this agent uses in the conversation (public arguments). We introduce the following sets:

$Support(Ag, p) = \{p' / p' \therefore p\}$
$Create\_Support(Ag_1, SC(Ag_1, Ag_2, t, p)) = \{SC(Ag_1, Ag_2, t_x, p_x) / p_x \therefore p\}$

$Support(Ag, p)$ is the set of *Ag*s' private arguments supporting *p*.
$Create\_Support(Ag_1, SC(Ag_1, Ag_2, t, p))$ is the set of commitments *created* by agent Ag_1 to support the content of $SC(Ag_1, Ag_2, t, p)$. This set is closed under the support relation i.e.:

$(SC(Ag_1, Ag_2, t_2, p_2) \in Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_1, p_1))$
$\wedge\ SC(Ag_1, Ag_2, t_1, p_1) \in Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_0, p_0)))$
$\Rightarrow SC(Ag_1, Ag_2, t_2, p_2) \in Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_0, p_0))$

We use the notation: p ⌂ $Arg\_Sys(Ag_1)$ to denote the fact that a propositional formula p can be generated from the argumentation system of Ag_1 denoted $Arg\_Sys(Ag_1)$. The formula $\neg(p$ ⌂ $Arg\_Sys(Ag_1))$ indicates the fact that p cannot be generated from Ag_1's argumentation system. A propositional formula p can be generated from an agent's argumentation system, if this agent can find an argument that supports p. To simplify the formalism, we use the notation $Act'(Ag_x, SC(Ag_i, Ag_j, t_0, p))$ to indicate the action that agent Ag_x performs on the commitment $SC(Ag_i, Ag_j, t_0, p)$ or on its content ($Act' \in \{Create, Withdraw, Accept\text{-}content, Challenge\text{-}content, Refuse\text{-}content\}$). For the actions related to the argumentation relations, we write $Act\text{-}Arg(Ag_x, [SC(Ag_n, Ag_m, t_1, q)], SC(Ag_i, Ag_j, t_0, p))$. This notation indicates that Ag_x defends (resp. attacks or justifies) the content of $SC(Ag_i, Ag_j, t_0, p)$ by the content of $SC(Ag_n, Ag_m, t_1, q)$ ($Act\text{-}Arg \in \{Defend\text{-}content, Attack\text{-}content, Justify\text{-}content\}$). The commitment that is written between square brackets [ ] is the support of the argument. In a general way, we use the notation $Act'(Ag_x, S)$ to indicate the action that Ag_x performs on the set of commitments S or on the contents of these commitments, and the notation $Act\text{-}Arg(Ag_x, [S], SC(Ag_i, Ag_j, t_0, p))$ to indicate the argumentation-related action that Ag_x performs on the content of $SC(Ag_i, Ag_j, t_0, p)$ using the contents of S as support. We also introduce the notation $Act\text{-}Arg(Ag_x, [S], S')$ to indicate that Ag_x performs an argumentation-related action on the contents of a set of commitments S' using the contents of S as supports.

We distinguish two types of dialogue games: *entry game* and *chaining games*. The entry game allows the two agents to *open* the persuasion dialogue. It corresponds to the entry conditions. The chaining games make it possible to continue the conversation. The protocol terminates when the exit conditions are satisfied (Figure 9.2).

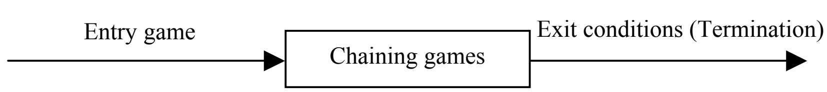
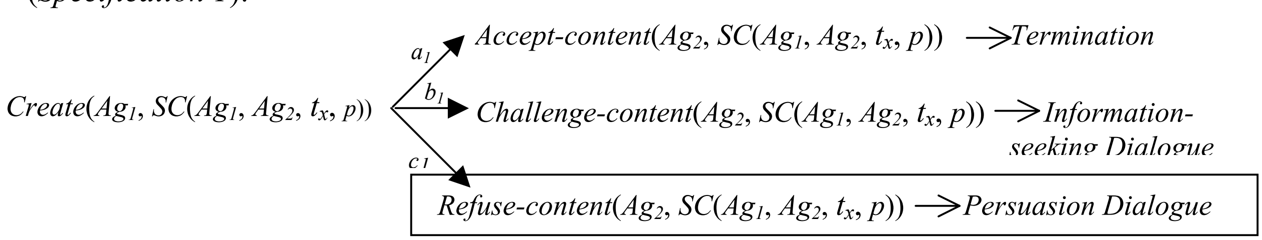


Figure 9.2. The general form of the protocol

9.2.4 Dialogue Games

A Entry Game

The entry game that describes the entry conditions in our persuasion protocol about a propositional formula p is described by the entry conversation policies as follows (*Specification 1*):

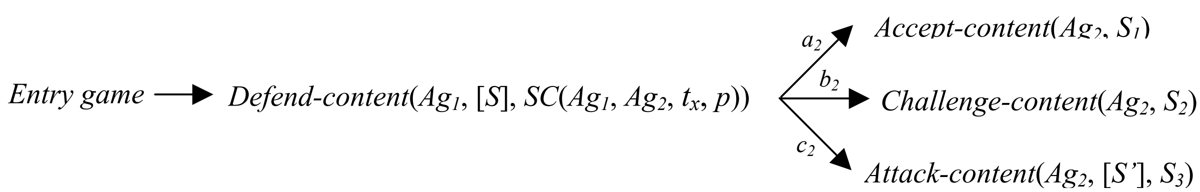


where a_1, b_1 and c_1 are three conditions specified as follows:
$a_1 = p \mathbin{⌂} Arg\_Sys(Ag_2)$
$b_1 = \neg(p \mathbin{⌂} Arg\_Sys(Ag_2)) \wedge \neg(\neg p \mathbin{⌂} Arg\_Sys(Ag_2))$
$c_1 = \neg p \mathbin{⌂} Arg\_Sys(Ag_2)$

If Ag_2 has an argument for p then it accepts p (the content of $SC(Ag_1, Ag_2, t_x, p)$) and the conversation terminates as soon as it begins (Condition a_1). If Ag_2 has neither an argument for p nor for $\neg p$, then it challenges p and the two agents open an information-seeking dialogue (condition b_1). The persuasion dialogue starts when Ag_2 refuses p because it has an argument against p (condition c_1).

B Defense Game

Once the two agents opened a persuasion dialogue, the initiator must defend its point of view. Thus, it must play a defense game. Our protocol is specified in such a way that the *persuasion dynamics* starts by playing a defense game. We have (*Specification 2*):



where:
$S = \{SC(Ag_1, Ag_2, t_i, p_i) / i = 0,\ldots,n\}$, p_i are commitment-free formulae.
$\bigcup_{i=1}^{3} S_i = S$, $S_i \cap S_j = \varnothing$, $i, j = 1,\ldots,3 \,\&\, i \neq j$

By definition, *Defend-content*$(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p))$ means that Ag_1 creates S in order to defend the content of $SC(Ag_1, Ag_2, t_x, p)$. Formally:

$$Defend\text{-}content(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p)) =_{def} (Create(Ag_1, S) \wedge S = Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_x, p)))$$

We consider this definition as an *assertional* description of the *Defend* action.

This specification indicates that according to the three conditions (a_2, b_2 and c_2), Ag_2 can accept a subset S_1 of S, challenge a subset S_2 and attack a third subset S_3. Sets S_1, S_2, and S_3 are mutually disjoint because Ag_2 cannot, for example, both accept and challenge the same commitment content. *Accept*, *Challenge* and *Attack* a set of commitment contents are defined as follows by the following formulae:

$$Accept\text{-}content(Ag_2, S_1) =_{def} (\forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_1 \Rightarrow Accept\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_i, p_i)))$$

$$Challenge\text{-}content(Ag_2, S_2) =_{def} (\forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_2 \Rightarrow Challenge\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_i, p_i)))$$

$$Attack\text{-}content(Ag_2, [S'], S_3) =_{def} \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_3 \Rightarrow \exists S'_j \subseteq S': Attack\text{-}content(Ag_2, [S'_j], SC(Ag_1, Ag_2, t_i, p_i))$$

where: $\bigcup_{j=0}^{m} S'_j = S'$.
This indication means that any element of S' is used to attack one or more elements of S_3.

The conditions a_2, b_2 and c_2 are specified as follows:

$a_2 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_1 \Rightarrow p_i \frown Arg\_Sys(Ag_2)$
$b_2 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_2 \Rightarrow (\neg(p_i \frown Arg\_Sys(Ag_2)) \wedge \neg(\neg p_i \frown Arg\_Sys(Ag_2)))$
$c_2 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_3 \Rightarrow \exists S'_j \subseteq S', Content(S'_j) = Support(Ag_2, \neg p_i)$
where $Content(S'_j)$ indicates the set of contents of the commitments S'_j.

C Challenge Game

The challenge game is specified as follows (*Specification 3*):

$$Challenge\text{-}content(Ag_1, SC(Ag_2, Ag_1, t_x, p)) \xrightarrow{a_3} Justify\text{-}content(Ag_2, [S], SC(Ag_2, Ag_1, t_x, p))$$

where the condition a_3 is specified as follows:

$a_3 = (Content(S) = Support(Ag_2, p))$

In this game, the condition a_3 is always true. The reason is that in accordance with the commitment semantics, an agent must always be able to defend the commitment it created (see Chapter 7).

D Justification Game

For this game we distinguish two cases:

Case1. $SC(Ag_1, Ag_2, t_x, p) \notin S$
In this case, Ag_1 justifies the content of its commitment $SC(Ag_1, Ag_2, t_x, p)$ by creating a set of commitments S. As for the Defend action, Ag_2 can accept, challenge and/or attack a subset of S. The specification of this case is given by the following conversation policies (*Specification 4*):

$$Justify\text{-}content(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p)) \xrightarrow{a_4} Accept\text{-}content(Ag_2, S_1)$$
$$Justify\text{-}content(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p)) \xrightarrow{b_4} Challenge\text{-}content(Ag_2, S_2)$$
$$Justify\text{-}content(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p)) \xrightarrow{c_4} Attack\text{-}content(Ag_2, [S'], S_3)$$

where:

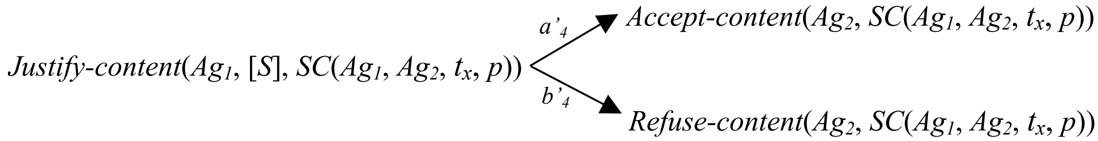
$S = \{SC(Ag_1, Ag_2, t_i, p_i) / i = 0,...,n\}$, p_i are propositional formulae.

$\bigcup_{i=1}^{3} S_i = S, S_i \cap S_j = \varnothing, i, j = 1,...3 \; \& \; i \neq j$

$a_4 = a_2,\ b_4 = b_2,\ c_4 = c_2$

Case2. $\{SC(Ag_1, Ag_2, t_x, p)\} = S$

In this case, the justification game has the following specification (*Specification 5*):



where:

$a_4' = Ag_1 \in Trust(Ag_2, D)$

$b_4' = Ag_1 \notin Trust(Ag_2, D)$

$Trust(Ag, D)$ is the set of the trustworthy agents for Ag relative to a domain D. Here we assume that p is in the domain D. This aspect will be discussed later.

Ag_1 justifies the content of its commitment $SC(Ag_1, Ag_2, t_x, p)$ by itself (i.e. by p). This means that p is part of Ag_1's knowledge. Only two moves are possible for Ag_2: 1) *accept* the content of $SC(Id_x, Ag_1, Ag_2, p)$ if Ag_1 is a trustworthy agent for Ag_2 (a'_4), 2) if not, *refuse* this content (b'_4). Ag_2 cannot attack this content because it does not have an argument against p. The reason is that Ag_1 plays a justification game because Ag_2 played a challenge game.

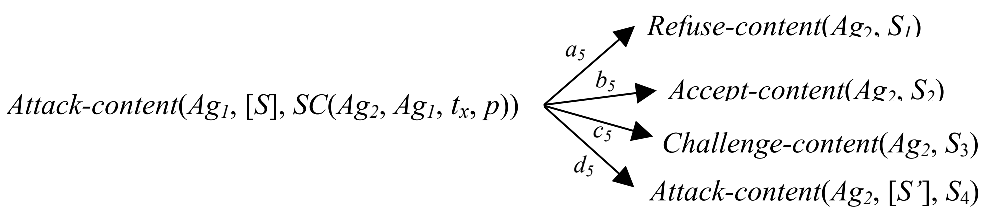
Like the definition of the *Defend* action, we define the *Justify* action as follows:

$$Justify\text{-}content(Ag_1, [S], SC(Ag_1, Ag_2, t_x, p)) =_{def} (Create(Ag_1, S) \wedge S = Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_x, p)))$$

This means that Ag_1 creates the set S of commitments to support the commitment $SC(Ag_1, Ag_2, t_x, p)$.

E Attack Game

The attack game is specified by the following conversation policies (*Specification 6*):



where:

$S = \{SC(Ag_1, Ag_2, t_i, p_i) / i = 0,...,n\}$, p_i are propositional formulae.

$\bigcup_{i=1}^{4} S_i = S, Card(S_1) = 1, S_i \cap S_j = \emptyset, i, j = 1,...,4 \;\&\; i \neq j$

Formally, the *Attack* action is defined as follows:

$$\begin{aligned}Attack\text{-}content(Ag_1, [S], SC(Ag_2, Ag_1, t_x, p)) =_{def} \exists t_y, (&Create(Ag_1, SC(Ag_1, Ag_2, t_y, \neg p)) \\ &\wedge Create(Ag_1, S) \\ &\wedge S = Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_y, \neg p)))\end{aligned}$$

This means that by attacking $SC(Ag_2, Ag_1, t_y, p)$, Ag_1 creates the commitment $SC(Ag_1, Ag_2, t_y, \neg p)$ and the set S to support this commitment.

The conditions a_5, b_5, c_5 and d_5 are specified as follows:

$a_5 = \exists i, t_z$: $SC(Ag_2, Ag_1, t_i, p_i) \in Create\_Support(Ag_2, SC(Ag_2, Ag_1, t_z, \neg q))$
where $S_1 = \{SC(Ag_1, Ag_2, t_{z'}, q)\}$
$b_5 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_2 \Rightarrow p_i \vartriangle Arg\_Sys(Ag_2)$
$c_5 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_3 \Rightarrow (\neg(p_i \vartriangle Arg\_Sys(Ag_2)) \wedge \neg(\neg p_i \vartriangle Arg\_Sys(Ag_2)))$
$d_5 = \forall i, SC(Ag_1, Ag_2, t_i, p_i) \in S_4 \Rightarrow \exists S'_j \subseteq S'$: $Content(S'_j) = Support(Ag_2, \neg p_i)$
$\wedge \exists t_z, \nexists k$: $SC(Ag_2, Ag_1, t_k, p_k) \in Create\_Support(Ag_2, SC(Ag_2, Ag_1, t_z, \neg p_i))$

Ag_2 refuses Ag_1's argument if Ag_2 already attacked this argument. In other words, Ag_2 refuses Ag_1's argument if Ag_2 cannot attack this argument since it *already* attacked it, and it cannot accept it or challenge it since it has an argument against this argument. We have only one element in S_1 because we consider a refusal move as an exit condition. The acceptance and the challenge actions of this game are the same as the acceptance and the challenge actions of the defense game. Finally, Ag_2 attacks Ag_1's argument if Ag_2 has an argument against Ag_1's argument, and if Ag_2 did not attack Ag_1's argument before. In d_5, the universal quantifier means that Ag_2 attacks all Ag_1's arguments for which it has an against-argument. The reason is that Ag_2 must act on all commitments created by Ag_1. The temporal aspect (the past) of a_5 and d_5 is implicitly integrated in $Create\_Support(Ag_2, SC(Ag_2, Ag_1, t_z, \neg q))$ and $Create\_Support(Ag_2, SC(Ag_2, Ag_1, t_i, \neg p_i))$.

F Termination Game

The protocol terminates either by a final acceptance or by a refusal. There is a final acceptance when Ag_2 accepts the content of the initial commitment $SC(Ag_1, Ag_2, t_x, p)$ or when Ag_1 accepts the content of $SC(Id_y, Ag_2, Ag_1, \neg p)$. Ag_2 accepts the content of $SC(Ag_1, Ag_2, t_x, p)$ iff it accepts all the supports of $SC(Ag_2, Ag_1, t_x, p)$. Formally:

$$\begin{aligned}&Accept\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_x, p)) \Leftrightarrow \\ &\quad (\forall i, SC(Ag_1, Ag_2, t_i, p_i) \in Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_x, p)) \\ &\quad \Rightarrow Accept\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_i, p_i)))\end{aligned}$$

The acceptance of the supports of $SC(Ag_1, Ag_2, t_x, p)$ by Ag_2 does not mean that they are accepted directly after their creation by Ag_1, but it can be accepted after a number of challenge, justification and attack games. When Ag_2 accepts definitively, then it withdraws all commitments whose content was attacked by Ag_1. Formally:

$$Accept\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_x, p)) \Rightarrow$$
$$(\forall i, \forall S, Attack\text{-}content(Ag_1, [S], SC(Ag_2, Ag_1, t_i, p_i))$$
$$\Rightarrow Withdraw(Ag_2, SC(Ag_2, Ag_1, t_i, p_i)))$$

On the other hand, Ag_2 refuses the content of $SC(Ag_1, Ag_2, t_x, p)$ iff it refuses one of the supports of $SC(Id_x, Ag_1, Ag_2, p)$. Formally:

$$Refuse\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_x, p)) \Leftrightarrow$$
$$(\exists i: SC(Ag_1, Ag_2, t_i, p_i) \in Create\_Support(Ag_1, SC(Ag_1, Ag_2, t_x, p))$$
$$\wedge\ Refuse\text{-}content(Ag_2, SC(Ag_1, Ag_2, t_i, p_i)))$$

9.2.5 Protocol Dynamics

The persuasion dynamics is described by the chaining of a finite set of dialogue games: acceptance move, refusal move, defense, challenge, attack and justification games. These games can be combined in a sequential and parallel way (Figure 9.3).

After Ag_1's defense game at moment t_1, Ag_2 can, at moment t_2, accept a part of the arguments presented by Ag_1, challenge another part, and/or attack a third part. These games are played in parallel. At moment t_3, Ag_1 answers the challenge game by playing a justification game and answers the attack game by playing an acceptance move, a challenge game, another attack game, and/or a final refusal move. The persuasion dynamics continues until the exit conditions become satisfied (final acceptance or a refusal). From our specifications, it follows that our protocol plays the role of the dialectical proof theory of the argumentation system.

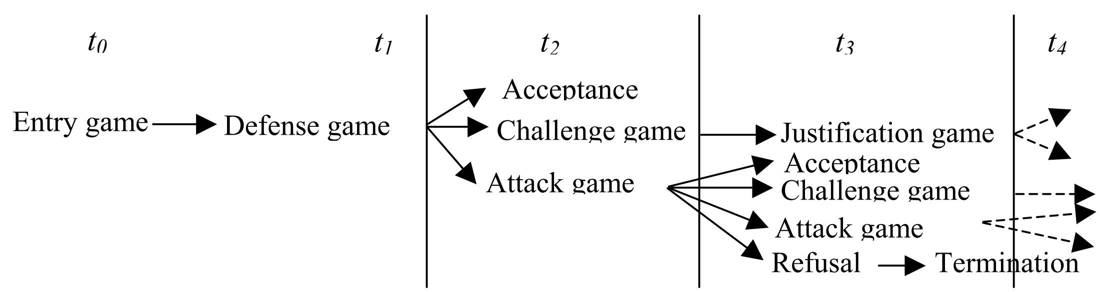


Figure 9.3. The persuasion dialogue dynamics

Indeed, our persuasion protocol can be described by BNF grammar. To do this, we first introduce the following definitions:

Let G_1, G_2, and G_3 three dialogue games.
$G_1 //_{\geq 1} G_2 = G_1 \mid G_2 \mid G_1 // G2$

$G_1 //_{opt} G_2 = \varepsilon \mid G_1 //_{\geq 1} G_2$
$//(G_1, G_2, G_3) = (G_1 //_{\geq 1} G_2) //_{opt} G_3$
$\mid (G_1 //_{opt} G_2) //_{\geq 1} G_3$

where: ε is the empty dialogue game, and "//" is the parallelization symbol. G_1 // G2 means that an agent can play the two games in parallel.

The persuasion protocol can be defined as follows:

Persuasion protocol = *Entry game* ; *Defense game* ; *Dialogue games*
Dialogue games = //(*Acceptance move* ; *Ch* ; *Att*)
Ch = *Challenge game* ; *Justification game* ; (*Dialogue games* | *Refusal*)
Att = *Attack game* ; (*Dialogue games* | *Refusal*)

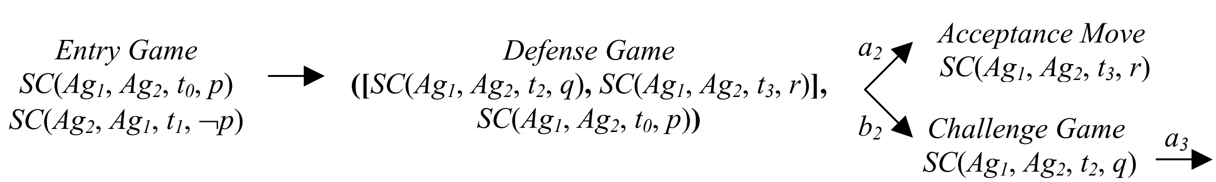
where ";" is the sequencing symbol.

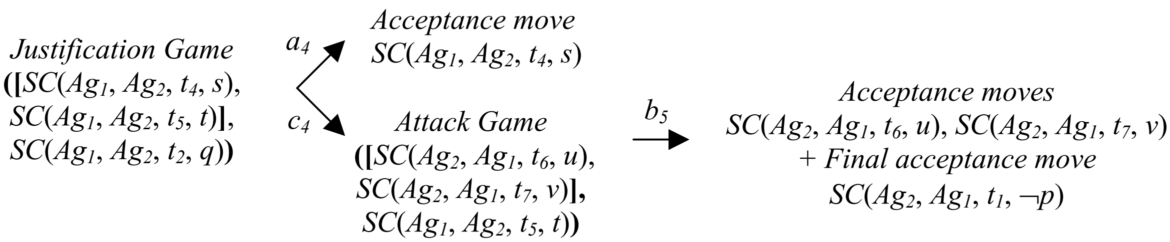
Example

In this section we present a simple example dialogue that illustrates some notions presented in this chapter.

Ag_1: Newspapers can publish information I (*p*).
Ag_2: I don't agree with you.
Ag_1: They can publish information I because it is not private (*q*), and any public information can be published (*r*).
Ag_2: Why is information I public?
Ag_1: Because it concerns a Minister (*s*), and information concerning a Minister is public (*t*).
Ag_2: Information concerning a Minister is not necessarily public, because information I is about the health of Minister (*u*), and information about the health remains private (*v*).
Ag_1: I accept your argument.

This example was also studied in (Amgoud and Maudet, 2002) in a context of strategical considerations for argumentative agents. The letters on the left of the utterances are the propositional formulae that represent the propositional contents. Agent Ag_1's KB contains: $([q, r], p)$, $([s, t], q)$ and $([u], u)$. Agent Ag_2's KB contains: $([\neg t], \neg p)$, $([u, v], \neg t)$, $([u], u)$ and $([v], v)$. The combination of the dialogue games that allows us to describe the persuasion dialogue dynamics is as follows:





Ag_1 creates $SC(Ag_1, Ag_2, t_0, p)$ to achieve the goal of persuading Ag_2 that p is true. Ag_1 can create this commitment because it has an argument for p. Ag_2 refuses $SC(Ag_1, Ag_2, t_0, p)$ because it has an argument against p. Thus, the entry game is played and the persuasion dialogue is opened. Ag_1 defends $SC(Ag_1, Ag_2, t_0, p)$ by creating $SC(Ag_1, Ag_2, t_2, q)$ and $SC(Ag_1, Ag_2, t_3, r)$. Ag_2 accepts $SC(Ag_1, Ag_2, t_3, r)$ because it has an argument for r and challenges $SC(Ag_1, Ag_2, t_2, q)$ because it has no argument for q or against q. Ag_1 plays a justification game to justify $SC(Ag_1, Ag_2, t_2, q)$ by creating $SC(Ag_1, Ag_2, t_4, s)$ and $SC(Ag_1, Ag_2, t_5, t)$. Ag_2 accepts the content of $SC(Ag_1, Ag_2, t_4, s)$ and attacks the content of $SC(Ag_1, Ag_2, t_5, t)$ by creating $SC(Ag_2, Ag_1, t_6, u)$ and $SC(Ag_2, Ag_1, t_7, v)$. Finally, Ag_1 plays acceptance moves because it has an argument for u and it does not have arguments against v and the dialogue terminates. Indeed, before accepting v, Ag_1 challenges it and Ag_2 defends it by itself (i.e. ([$SC(Ag_2, Ag_1, t_7, v)$, $SC(Ag_2, Ag_1, t_7, v)$])). Then, Ag_1 accepts this argument because it considers Ag_2 trustworthy (see Figure 9.9 Section 9.4). Ag_1 updates its KB by removing the attacked argument and including the new argument. Figure 9.12 (Section 9.4) illustrates the screen shot of this example generated by our prototype. In this figure commitments are described only by their contents and the identifiers of the two agents are the two first arguments of the exchanged communicative actions. The contents are specified using a predicate language that the two agents share (the ontology).

9.2.6 Algorithms

The general algorithm representing our persuasion dialogue game protocol is given by Algorithm 9.1. Part *A* of Algorithm 9.1 specifies the entry conditions. Part *B* indicates the exit conditions. The persuasion dynamics (i.e. the sequence of utterances) is given by the function *Dynamics*. The specification of this function is given by Algorithms 9.2, 9.3, 9.4, 9.5 and 9.6. To simplify these algorithms, we suppose that the support of an argument is composed only by one commitment. In these algorithms S_{Ag1} indicates the set of arguments of agent Ag_1 (i.e. its knowledge base). S'_{Ag1} indicates the set of arguments that Ag_1 used in the current dialogue. The set S'_{Ag1} allows us to avoid the use of same arguments several times. These algorithms specify the different dialogue games of our protocol as *if then* rules.

```
{ If FΩ(Ag1, SC(Idx, Ag1, Ag2, p)) = {(Create, ti)}                    }
    And FΩ(Ag2, SC(Idx, Ag1, Ag2, p)) = {(Refuse-content, ti+1)}       } Part A
      Then
        { Conflict := 1;
          Dynamics;
         If Conflict  = 0 Then                         }
            " The conflict is resolved "               } Part B
        Else " The conflict is not resolved "          }
        }
}
```

Algorithm 9.1

Algorithm 9.2 deals with the acceptance (Termination game) and the refusal (Entry game) cases. The acceptance of $SC(Id_x, Ag_1, Ag_2, p)$ makes it possible to solve the conflict and to stop the algorithm. In the refusal case, if Ag_1 finds an argument (r, q) not yet used for its commitment $SC(Id_y, Ag_1, Ag_2, q)$, then this agent creates a new commitment $SC(Id_z, Ag_1, Ag_2, r)$ to defend $SC(Id_y, Ag_1, Ag_2, q)$. Ag_1 updates the set S'_{Ag1} by adding the argument (r, q). Ag_1 informs Ag_2 about its action using the *Send* primitive. The *Send* primitive has the form *Send*(*Destination*, *Action*). If Ag_1 does not have arguments to defend its commitment, then the conflict cannot be solved because each agent refuses the arguments of the other and the algorithm stops.

```
If (Accept-content, tj) ∈ FΩ(Ag2, SC(Ag1, Ag2, tx, p)) Then {
      Conflict := 0;
      Return Conflict;
 }
If FΩ(Ag2, SC(Ag1, Ag2, ty, q)) = {(Refuse-content, tj)} Then {
   If ∃(r, q) ∈ SAg1 / S'Ag1 Then {
        FΩ(Ag1, SC(Ag1, Ag2, tz, r)) := {(Create, tj+1);
        FEΣ(SC(Ag1, Ag2, tz, r), SC(Ag1, Ag2, ty, q)) := (Defend-content, tj+1) ;
        S'Ag1 := S'Ag1 ∪ {(r, q)};
        Send(Ag2, Defend(SC(Ag1, Ag2, tz, r), SC(Ag1, Ag2, ty, q)));
    }
     Else {
        Conflict : = -1;
        Return Conflict;
     }
 }
```

Algorithm 9.2

Algorithm 9.3 deals with the Challenge game. Ag_1 justifies its commitment if it finds an argument not yet used. As for the refusal case, Ag_1 updates S'_{Ag1} and informs Ag_2 about its action. If Ag_1 does not find such an argument, then it indicates to Ag_2 that the content of the challenged commitment is knowledge that Ag_1 believes true by justifying it by itself. The formal definition of the *justification* relation is the same as the *defense* relation.

```
If FΩ(Ag2, SC(Ag1, Ag2, ty, q)) = {(Challenge-content, tj)} Then {
   If ∃(r, q) ∈ SAg1 / S'Ag1 Then {
     FΩ(Ag1, SC(Ag1, Ag2, tz, r)) : = {(Create, tj+1)};
     FEΣ(SC(Ag1, Ag2, tz, r), SC(Ag1, Ag2, ty, q)) := (Justify-content, tj+1);
     S'Ag1 = S'Ag1 ∪ {(r, q)};
     Send(Ag2, Justify(SC(Ag1, Ag2, tz, r), SC(Ag1, Ag2, ty, q)));
   }
   Else {
      FEΣ(SC(Ag1, Ag2, ty, q), SC(Ag1, Ag2, ty, q)) := (Justify-content, tj+1);
      Send(Ag2, Justify(SC(Ag1, Ag2, ty, q), SC(Ag1, Ag2, ty, q)));
   }
}
```

Algorithm 9.3

Algorithm 9.4 deals with the case of Ag_1 reaction if Ag_2 justifies the content of its commitment by itself (case 2 of Justification game). *Trustworthy*(Ag_2, q) is a boolean function that enables Ag_1 to determine if Ag_2 is trustworthy or not. If according to Ag_1, Ag_2 is trustworthy, then Ag_1 accepts Ag_2's commitment. If not, Ag_1 refuses Ag_2's commitment. In the next section (Section 9.3) we propose a probabilistic model of trustworthiness to determine the value of *Trustworthy*(Ag_2, q) function.

```
If FEΣ(SC(Ag2, Ag1, ty, q), SC(Ag2, Ag1, ty, q)) = (Justify-content, tj) Then {
     If Trustworthy(Ag2, q)
        Then FΩ(Ag1, SC(Ag2, Ag1, ty, q)) := {(Accept-content, tj+1)}
     Else FΩ(Ag1, SC(Ag2, Ag1, ty, q)) := {(Refuse-content, tj+1)}
    Send(Ag2, Refuse(Ag1, SC(Ag2, Ag1, ty, q)));
  }
```

Algorithm 9.4

Algorithm 9.5 deals with the case where Ag_2 attacks the support of Ag_1's argument (Attack game). Ag_1 attacks Ag_2's argument if Ag_1 has an against-argument not already used. If not Ag_1 refuses this argument. If Ag_1 cannot attack or refuse Ag_2's argument, then Ag_1 accepts

Ag_2's argument if *Ag_1* has an argument. If not *Ag_1* challenges *Ag_2*'s argument if *Ag_1* has no arguments nor against-arguments.

```
If F_EΣ(SC(Ag_2, Ag_1, t_y, q), SC(Ag_1, Ag_2, t_z, r)) = (Attack-content, t_j) Then {
    If ∃(s, ¬q) ∈ S_Ag1 / S'_Ag1 Then {
      F_Ω(Ag_1, SC(Ag_1, Ag_2, t_z', s)) := {(Create, t_j+1)};
      F_EΣ (SC(Ag_1, Ag_2, t_z', s), SC(Ag_2, Ag_1, t_y, q)) := (Attack-content, t_j+1);
      S'_Ag1 := S'_Ag1 ∪ {( s, ¬q)};
      Send(Ag_2, Attack(SC(Ag_1, Ag_2, t_z', s), SC(Ag_2, Ag_1, t_y, q)));
    }
Else
        If ∃(s, ¬q) ∈ S'_Ag1 then
          F_Ω(Ag_1, SC(Ag_2, Ag_1, t_y, q)) = {(Refuse-content, t_j)}
        Else {
           If (s, q) ∈ S_Ag1 / S'_Ag1 Then
              F_Ω(Ag_1, SC(Ag_2, Ag_1, t_y, q)) := {(Accept-content, t_j+1)}
           Else F_Ω(Ag_1, SC(Ag_2, Ag_1, t_y, q)) := {(Challenge-content, t_j+1)};
           Send(Ag_2, Challenge(Ag_1, SC(Id_y, Ag_2, Ag_1, q)));
          }
  }
```

Algorithm 9.5

Algorithm 9.6 deals with the case in which the reactive game of *Ag_2* is a defense of its argument (Defense game) or a justification of its commitment (case 1 of Justification game). Thus, *Ag_1* can attack the support of the *Ag_2*'s argument or its conclusion according to *Ag_1*'s arguments. As in Algorithm 9.5, *Ag_1* accepts or challenges the support of *Ag_2*'s argument in the opposite case.

9.2.7 Termination Proof

In this section we discuss the termination of our protocol (i.e. the termination of Algorithm 9.1). Informally, to prove the termination of Algorithm 9.1, it is enough to prove that the protocol dynamics always converges to a final acceptance or a final refusal.

According to the Algorithms 9.2, 9.3, 9.4, 9.5 and 9.6, the protocol chaining can have one of the following possibilities:

1- Agent *Ag_2* accepts all the supports of the initial commitment $SC(Ag_1, Ag_2, t_x, p)$. Therefore, we have: $(Accept\text{-}content, t_i) \in F_{\Omega}(Ag_2, SC(Ag_1, Ag_2, t_x, p))$.

2- Agent *Ag_2* refuses one of the supports of $SC(Id_x, Ag_1, Ag_2, p)$, and *Ag_1* does not find an argument to defend this support. Thus, we have: $F_{\Omega}(Ag_2, SC(Id_x, Ag_1, Ag_2, p)) = \{\ldots, (Refuse\text{-}content, t_i)\}$.

3- The two agents attack each other about a part of the last arguments.

4- Agent Ag_2 challenges a part of the arguments presented by Ag_1.

```
If FEΣ(SC(Ag2, Ag1, ty, q), SC(Ag2, Ag1, tz, r)) = (Defend-content, tj) or
  FEΣ(SC(Ag2, Ag1, ty, q), SC(Ag2, Ag1, tz, r)) = (Justify-content, tj) Then {
    If ∃(s, ¬q) ∈ SAg1 / S'Ag1 Then {
      FΩ(Ag1, SC(Ag1, Ag2, tz', s)) := {(Create, tj+1)};
      FEΣ(SC(Ag1, Ag2, tz', s), SC(Ag2, Ag1, ty, q)) := (Attack-content, tj+1);
      S'Ag1 := S'Ag1 ∪ {(s, ¬q)};
      Send(Ag2, Atack(SC(Ag1, Ag2, tz', s), SC(Ag2, Ag1, ty, q)));
    }
    Else
     If ∃(s, ¬r) ∈ SAg1 / S'Ag1 Then {
      FΩ(Ag1, SC(Ag1, Ag2, tz', s)) := {(Create, tj+1)};
      FEΣ(SC(Ag1, Ag2, tz', s), SC(Ag2, Ag1, tz, r)) := (Attack-content, tj+1);
      S'Ag1 := S'Ag1 ∪ {(s, ¬r)};
      Send(Ag2, Attack(SC(Ag1, Ag2, tz', s), SC(Ag2, Ag1, tz, r)));
     }
     Else {
          If ∃(s, q) ∈ SAg1 / S'Ag1 Then
            FΩ(Ag1, SC(Ag2, Ag1, ty, q)) := {(Accept, tj+1)};
          Else FΩ(Ag1, SC(Ag2, Ag1, ty, q)) := {(Challenge, tj+1)};
          Send(Ag2, Challenge(Ag1, SC(Ag2, Ag1, ty, q)));
     }
}
```

Algorithm 9.6

Possibilities 1 and 2 converge to a final acceptance and a final refusal. Possibility 3 converges to a situation where an agent finds an argument (*H*, *h*) to attack the support of the interlocutor's argument, but this argument was already used ((*H*, *h*) ∈ S'_{Ag}). The reason is that the agents' knowledge bases are finite. In this case, this agent refuses the interlocutor's argument (Algorithm 9.2). Thus, possibility 3 converges to a final refusal. For the same reason, possibility 4 converges to the situation in which Ag_1 justifies a support by itself. In this situation, Ag_2 can play only an acceptance move if Ag_1 is trustworthy or a refusal move if not (Algorithm 9.4). Thus, possibility 4 converges to a final acceptance or a final refusal.

Formally, the termination of our dialogue game protocol is stated by the following theorem.

Theorem 9.1 *The protocol dynamics always terminates.*

Proof
To prove this theorem, we use a tableau method (Cleaveland, 1990). The idea is to formalize our specifications (Section 9.2.4) as tableau rules and then to prove the finiteness of the tableau. Tableau rules are written in such a way that premises appear above conclusions. Using a tableau method means that the specifications are conducted in a top-down fashion. For example, *specification 2* (defense game) can be expressed by the following rules:

$$R1: \frac{Defend(Ag_1,[S],SC(p))}{Accept(Ag_2,S_1)}$$

$$R2: \frac{Defend(Ag_1,[S],SC(p))}{Challenge(Ag_2,S_1)}$$

$$R3: \frac{Defend(Ag_1,[S],SC(p))}{Attack(Ag_2,[S'],S_1)}$$

We denote the formulae of our specifications by σ, and we define E_σ the set of σ. We define an ordering $\prec$ on E_σ and we prove that $\prec$ has no infinite ascending chains. Intuitively, this relation is to hold between σ_1 and σ_2 if it is possible that σ_1 is an ancestor of σ_2 in some tableau. Before defining this ordering, we introduce some notations: *Act**(*Ag*, [*S*], *S'*) with *Act** $\in$ {*Act'*, *Act-Arg*} is a formula. We notice that formulae in which there is no support [*S*], can be written as follows: *Act**(*Ag*, [$\varnothing$], *S'*). $\sigma[S] \rightarrow_R \sigma[S']$ indicates that the tableau rule *R* has the formula $\sigma[S]$ as premise and the formula $\sigma[S']$ as conclusion, with $\sigma[S]$ = *Act**(*Ag*, [*S*], *S'*). The size $|S|$ is the number of commitments in *S*.

Definition 9.2 *Let* $\sigma[S_i]$ *be a formula and* E_σ *the set of* $\sigma[S_i]$. *The ordering* $\prec$ *on* E_σ *is defined as follows. We have* $\sigma[S_0] \prec \sigma[S_1]$ *if*:
$|S_1| < |S_0|$ *or*
For all rules Ri such that $\sigma[S_0] \rightarrow_{R0} \sigma[S_1] \rightarrow_{R1} \sigma[S_2] \ldots \rightarrow_{Rn} \sigma[S_n]$ *we have* $|S_n| = 0$.

Intuitively, in order to prove that a tableau system is finite, we need to prove the following:
1- if $\sigma[S_0] \rightarrow_R \sigma[S_1]$ then $\sigma[S_0] \prec \sigma[S_1]$.
2- $\prec$ has no infinite ascending chains (i.e. the inverse of $\prec$ is well-founded).

Property 1 reflects the fact that applying tableau rules results in shorter formulae, and property 2 means that this process has a limit. The proof of 1 proceeds by a case analysis on *R*. Most cases are straightforward. We consider here the case of *R*3. For this rule we have two cases. If $|S_1| < |S_0|$, then $\sigma[S_0] \prec \sigma[S_1]$. If $|S_1| \geq |S_0|$, we can apply the rules corresponding to the Attack game specification. The three first rules are straightforward since $S_2 = \varnothing$. For the last rule, we have the same situation that *R*3. Suppose that there is no path in the tableau $\sigma[S_0] \rightarrow_{R0} \sigma[S_1] \rightarrow_{R1} \sigma[S_2] \ldots \rightarrow_{Rn} \sigma[S_n]$ such that $|S_n| = 0$. This means that i) the number of arguments that agents have is infinite or that ii) one or several arguments are used several times. However, situation i is not possible because the agents' knowledge bases S_{Ag} are finite sets, and situation ii is not allowed in our protocol because agents cannot use arguments already used (i.e. arguments already in in S'_{Ag}). We note here that the agents' knowledge bases are updated after each conversation by removing the attacked arguments that cannot be defended and adding the new accepted arguments.

Because the definition of $\prec$ is based on the size of formulae and since $|S_0| \in N$ ($< \infty$) and $<$ is well-founded in $\mathbb{N}$, it follows that there is no infinite ascending chains of the form $\sigma[S_0] \prec \sigma[S_1]\ldots$

□

We notice that what we proved here is the termination of the protocol run and not the termination of the dialogue. For this reason, this proof uses the protocol specification in terms of the dialogue rules. It is clear that the termination of the protocol run results in the termination of the dialogue.

9.2.8 Correctness and Complexity

Correctness. We can formalize the correctness problem of our algorithms as follows: Algorithm 9.1 is correct iff the protocol description based on this algorithm satisfies the protocol specification (i.e. what the protocol must do). The specification can be formalized as a set of claims or properties that must be predefined. The idea is to describe the protocol as a transition system T for a dialogue game protocol as defined in Chapter 8 (Definition 8.2), and to express the specification as logical formulae ψ using our DCTL*$_{CAN}$ logic (see Chapter 7). This formalization enables us to deal with the correctness problem as a model-checking problem, i.e. whether $T \models \psi$ or not. For this purpose we can use our model checking technique that we proposed in Chapter 8.

Because our persuasion dialogue game protocol is specified using our pragmatic approach (Chapters 5 and 6), and dialogue game specifications are described as *if then* rules, it is easy to translate this protocol to a transition system T for a dialogue game. Transitions are labeled by the different actions that we use in our specifications of dialogue games (i.e. *Action_Ag$_i$*). The syntax of these actions can be easily translated to the syntax of DCTL*$_{CAN}$. For example the action:

Defend-content(Ag_1, [S], $SC(Id_x, Ag_1, Ag_2, p)$)

can be translated to:

Defend-content(Ag_1, $SC(Id_x, Ag_1, Ag_2, p)$, p')

where:
$S = \{SC(id_i, Ag_1, Ag_2, p_i) / i = 0,\ldots,n\}$ and $p' = p_0 \wedge p_1 \ldots \wedge p_n$.

Each dialogue game can be described by a fragment of the transition system T as follows: each conversation policy of the form :

$$Action\_Ag_1 \xrightarrow{Cond} Action\_Ag_2$$

can be described by two states s_1 and s_2 and a transition $s_1 \rightarrow^{Action\_Ag2} s_2$. $Action\_Ag_1$ is the label of a transition whose s_1 is the target state. We notice that the condition Cond is omitted. This does not affect the correctness of the protocol, because the conditions are used by agents as a reasoning mechanism about the protocol and do not belong to the protocol itself. Using this procedure, we can describe our persuasion protocol by a transition system for dialogue game protocol with 11 states and 16 transitions. The initial state s_0 is the source state of one transition labeled by the creation action. This transition system has two final states correspond respectively to the acceptance and the refusal states. Finally, the properties to be verified are derived from the specifications. The properties described in Chapter 8 (Section 8.4.2) are examples of the properties that our protocol must satisfied.

Complexity. The purpose of Algorithm 9.1 is to resolve the initial conflict or to decide after a finite number of moves that the conflict can not be resolved. Every move is based on the state of S_{Ag} and S'_{Ag} because agents must seek arguments or counter-arguments in S_{Ag} and S'_{Ag}. If we do not take into account the trustworthiness part of the algorithm, and since $|S_{Ag}| < |S'_{Ag}|$, the time complexity of Algorithm 9.1 is: $O(max(|S_{Ag1}|, |S_{Ag2}|))$. The complexity of the trustworthiness part will be discussed in Section 9.3.3.

9.3 Trustworthiness Model

9.3.1 Formulation

Several models of trustworthiness have been developed in the context of MAS (Sabater and Sierra, 2002), (Yu and Singh, 2002), (Ramchurn et al., 2003). However, their formulations do not take into account the elements we use in our approach (accepted and refused arguments, satisfied and violated commitments). For this reason, we propose a model that is more appropriate for our protocol. This model has the advantage of being simple and rigorous.

In our model, an agent's trustworthiness is a probability function defined as follows:

$$TRUST : A \times A \times D \rightarrow [0,1]$$

This function associates to each agent a probability measure representing its trustworthiness in the domain D according to another agent. Let X be a random variable representing an agent's trustworthiness. To evaluate the trustworthiness of an agent Ag_b, an agent Ag_a uses the records of its interactions with Ag_b. Equation 9.1 indicates how to calculate this trustworthiness as a probability measure (number of successful outcomes / total number of possible outcomes).

$$TRUST(Ag_b)_{Aga} = \frac{Nb\_\arg(Ag_b)_{Aga} + Nb\_SC(Ag_b)_{Aga}}{T\_Nb\_\arg(Ag_b)_{Aga} + T\_Nb\_SC(Ag_b)_{Aga}} \quad \textbf{(9.1)}$$

$Nb\_arg(Ag_b)_{Aga}$ is the number of Ag_bs' arguments that are accepted by Ag_a.
$Nb\_SC(Ag_b)_{Aga}$ is the number of satisfied commitments whose Ag_b is the debtor and Ag_a is the creditor.
$T\_Nb\_arg(Ag_b)_{Aga}$ *is the total number of* Ag_b*s' arguments towards* Ag_a.
$T\_Nb\_SC(Ag_b)_{Aga}$ is the total number of commitments whose Ag_b is the debtor and Ag_a is the creditor.

All these commitments and arguments are related to the domain *D*. The basic idea is that the trust degree of an agent can be induced according to how much information acquired from it has been accepted as belief in the past. Because all the factors of Equation 9.1 are related to the past, this information number is finite.

$TRUST(Ag_b)_{Aga}$ is the trustworthiness of Ag_b according to Ag_a's point of view. This trustworthiness is a dynamic value that changes according to the interactions taking place between Ag_a and Ag_b. This supposes that Ag_a knows Ag_b. If not, or if the number of interactions is not sufficient to determine this trustworthiness, the consultation of other agents becomes necessary.

As proposed in (Abdul-Rahman and Hailes, 2000) (Yu and Singh, 2002), each agent has two kinds of beliefs when evaluating the trustworthiness of another agent: local beliefs and total beliefs. Local beliefs are based on the direct interactions between agents. Total beliefs are based on the combination of the different testimonies of other agents called *witnesses*. In our model, local beliefs are given by Equation 9.1. Total beliefs require studying how different probability measures offered by witnesses can be combined. We deal with this aspect in the following section.

9.3.2 Estimating Agent's Trustworthiness

Let us suppose that an agent Ag_a wants to evaluate the trustworthiness of an agent Ag_b with which it never (or not enough) interacted before. This agent must consult agents that it knows to be trustworthy (*confidence agents*). A trustworthiness threshold *w* must be fixed. Thus, Ag_b will be considered trustworthy by Ag_a iff $TRUST(Ag_b)_{Aga}$ is higher or equal to *w*. Ag_a attributes a trustworthiness measure to each confidence agent Ag_i. When it is consulted by Ag_a, each confidence agent Ag_i provides a trustworthiness value for Ag_b if Ag_i knows Ag_b. Confidence agents use their local beliefs to calculate this value (Equation 9.1). Thus, the problem consists in evaluating Ag_b's trustworthiness using the trustworthiness values transmitted by confidence agents. Figure 9.4 illustrates this problem.

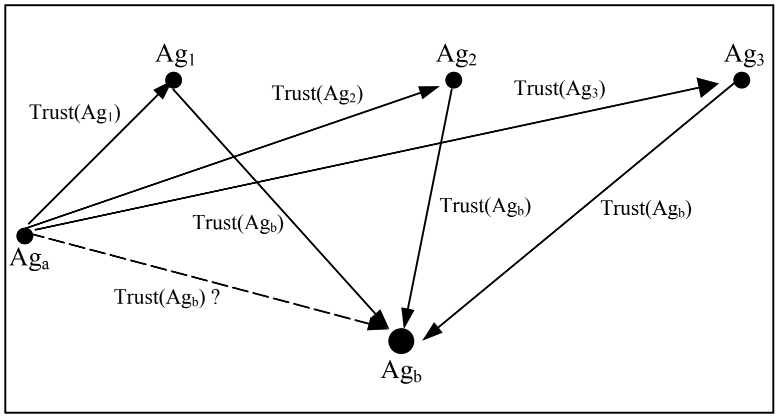


Figure 9.4. Problem of measuring Ag_b's trustworthiness by Ag_a

We notice that this problem cannot be formulated as a problem of conditional probability. Consequently, it is not possible to use *Bayes' theorem* or *total probability theorem*. The reason is that events in our problem are not mutually exclusive, whereas this condition is necessary for these two theorems. Probability values offered by confidence agents are not mutually exclusive since they are provided simultaneously.

To solve this problem we must study the distribution of the random variable X representing the trustworthiness of Ag_b. Since X takes only two values: 0 (the agent is not trustworthy) or 1 (the agent is trustworthy), variable X follows a Bernoulli distribution $\beta(1, p)$. According to this distribution, we have:

$$E(X) = p \qquad \textbf{(9.2)}$$

where $E(X)$ is the expectation of the random variable X *and* p is the probability that the agent is trustworthy. Thus, p is the probability that we seek. Therefore, *it is enough to calculate the expectation E(X) to find* $TRUST(Ag_b)_{Ag_a}$. However, this expectation is a theoretical mean that we must estimate. To this end, we can use the *Central Limit Theorem* (CLT) and the *law of large numbers*. The CLT states that whenever a random sample of size n $(X_1,\ldots X_n)$ is taken from any distribution with mean μ, then the sample mean $(X_1 + \ldots +X_n)/n$ will be approximately normally distributed with mean μ. As an application of this theorem, the arithmetic mean (average) $(X_1+\ldots+ X_n)/n$ approaches a normal distribution of mean μ, the expectation and standard deviation $\sigma/\sqrt{n}$ .Generally, and according to the law of large numbers, the expectation can be estimated by the weighted arithmetic mean.

Our random variable X is the weighted average of n independent random variables X_i that correspond to Ag_b's trustworthiness according to the point of view of confidence agents Ag_i. These random variables follow the same distribution: the Bernoulli distribution. They are also independent because the probability that Ag_b is trustworthy according to an agent Ag_t is independent of the probability that this agent (Ag_b) is trustworthy according to another agent Ag_r. Consequently, the random variable X follows a normal distribution whose average is the weighted average of the expectations of the independent random variables X_i. The estimation of expectation $E(X)$ can be given by Equation 9.3.

$$M_0 = \frac{\sum_{i=1}^{n} TRUST(Ag_i)_{Ag_a} TRUST(Ag_b)_{Ag_i}}{\sum_{i=1}^{n} TRUST(Ag_i)_{Aga}} \qquad \textbf{(9.3)}$$

The value M_0 represents an estimation of $TRUST(Ag_b)_{Aga}$.

Equation 9.3 does not take into account the number of interactions between confidence agents and Ag_b. This number is an important factor because it makes it possible to favor information coming from agents knowing more Ag_b. Equation 9.4 gives us an estimation of $TRUST(Ag_b)_{Aga}$ if we take into account this factor and we suppose that all confidence agents have the same trustworthiness.

$$M_1 = \frac{\sum_{i=1}^{n} N(Ag_i)_{Agb} TRUST(Ag_b)_{Ag_i}}{\sum_{i=1}^{n} N(Ag_i)_{Agb}} \qquad \textbf{(9.4)}$$

where $N(Ag_i)_{Agb}$ indicates the number of interactions between a confidence agent Ag_i and Ag_b. This number can be identified by the total number of Ag_b's commitments and arguments.

The combination of Equations 9.3 and 9.4 gives us a good estimation of $TRUST(Ag_b)_{Aga}$ (Equation 9.5) that takes into account the three most important factors: (1) the trustworthiness of confidence agents according to the point of view of Ag_a (2) the Ag_b's trustworthiness according to the point of view of confidence agents (3) the number of interactions between confidence agents and Ag_b. This number is an important factor because it makes it possible to favor information coming from agents knowing more Ag_b.

$$M_2 = \frac{\sum_{i=1}^{n} TRUST(Ag_i)_{Ag_a} N(Ag_i)_{Agb} TRUST(Ag_b)_{Ag_i}}{\sum_{i=1}^{n} TRUST(Ag_i)_{Aga} N(Ag_i)_{Agb}} \qquad \textbf{(9.5)}$$

This Equation shows how trust can be obtained by merging the trustworthiness values transmitted by some mediators. This merging method takes into account the proportional relevance of each trustworthiness value, rather than treating them equally. The function *Trustworthy*(Ag_2) of Algorithm 9.4 can be specified as follows:

If M > w Then Return true Else return false.

According to Equation 9.5, we have:

$$\forall i, TRUST(Ag_b)_{Agi} < w \Rightarrow M < w.\frac{\sum_{i=1}^{n} TRUST(Ag_i)_{Ag_a} N(Ag_i)_{Agb}}{\sum_{i=1}^{n} TRUST(Ag_i)_{Aga} N(Ag_i)_{Agb}}$$

$$\Rightarrow M < w$$

Consequently, the well-known *lottery paradox* of Kyburg can never happen. If all trustworthiness values transmitted by the mediators are below the threshold w, then Ag_a will not trust Ag_b.

To calculate M, we need the trustworthiness of other agents. A practical solution consists in building a *trust graph* like the *TrustNet* proposed by Yu and Singh (2002).

9.3.3 Trust Graph

In previous section (Section 9.3.2)) we offered a solution to the trustworthiness combination problem to evaluate the trustworthiness of a new agent (Ag_b). To simplify the problem we supposed that each consulted agent (a *confidence agent*) offers a trustworthiness value of Ag_b if it knows it. If a confidence agent does not offer any trustworthiness value, it will not be taken into account at the moment of the evaluation of Ag_b's trustworthiness by Ag_a. However, as outlined in (Yu and singh, 2002), a confidence agent can, if it does not know Ag_b, offer to Ag_a a set of agents which eventually know Ag_b. In this case, Ag_a will consult the proposed agents. These agents also have a trustworthiness value according to the point of view of the agent that proposed them. For this reason, Ag_a applies Equation 9.5 to assess the trustworthiness values of these agents. These new values will be used to evaluate the Ag_b's trustworthiness. We can build a trust graph in order to deal with this situation. Such a graph is defined as follows:

Definition 9.8 *A trust graph is a directed and weighted graph. The nodes are agents and an edge* (Ag_i, Ag_j) *means that agent* Ag_i *knows agent* Ag_j. *The weight of the edge* (Ag_i, Ag_j) *is a pair* (x, y) *where x is the* Ag_j*'s trustworthiness according to the point of view of* Ag_i *and y is the interaction number between* Ag_i *and* Ag_j. *The weight of a node is the agent's trustworthiness according to the point of view of the source agent.*

According to this definition, in order to determine the trustworthiness of the target agent Ag_b, it is necessary to find the weight of the node representing this agent in the graph. The graph is constructed while Ag_a receives answers from the consulted agents. The evaluation process of the nodes starts when all the graph is built. This means that this process only starts when Ag_a has received all the answers from the consulted agents. The process terminates when the node representing Ag_b is evaluated. The graph construction and the node evaluation algorithms are given respectively by Algorithms 9.7 and 9.8.

```
Construct-Graph(Aga, Agb, Limit_Nbr_Visited_Agents, Limit_Nbr_Witnesses)
{
  Graph := ∅
  Nbr_Witnesses := 0
  Nbr_Visited_Agents := 0
  Nbr_Additional_Agents :=
                    Max(0, Limit_Nbr_Visited_Agents – Size(Confidence(Aga)))
  Potential_Witnesses := Confidence(Aga)
  Add Node(Agb) to Graph

  While (Potential_Witnesses ≠ ∅) and
        (Nbr_Witnesses < Limit_Nbr_Witnesses) and
        (Nbr_Visited_Agents < Limit_Nbr_Visited_Agents) {

      n := Limit_Nbr_Visited_Agents - Nbr_Visited_Agents
      m := Limit_Nbr_Witnesses - Nbr_Witnesses

      For (i =1, i ≤ min(n, m), i++) {
          Ag1 := Potential_Witnesses(i)
          If Node(Ag1) ∉ Graph Then Add Node(Ag1) to Graph
          If Ag1 ∈ Confidence(Aga) Then Weight(Node(Ag1)) := Trust(Ag1)Aga
          Send(Ag1, Investigation(Agb))
          Nbr_Visited_Agents := Nbr_Visited_Agents +1 }

      For (i =1, i ≤ min(n, m), i++) {
          Ag1 := Potential_Witnesses(1)
          Str := Receive(Ag1)
          Potential_Witnesses := Potential_Witnesses / {Ag1}
          While (Str.Agents ≠ ∅) and (Nbr_Additional_Agents > 0) {
              If Str.Agents = {Agb} Then {
                  Nbr_Witnesses := Nbr_Witnesses + 1
                  Add Arc(Ag1, Agb)
                  Weight1(Arc(Ag1, Agb)) := Str.TRUST(Agb)Ag1
                  Weight2(Arc(Ag1, Agb)) := Str.n(Agb)Ag1
                  Str.Agents := ∅ }
              Else {
                  Nbr_Additional_Agents := Nbr_Additional_Agents – 1
                  Ag2 := Str.Agents(1)
                  Str.Agents := Str.Agents / {Ag2}
                  If Node(Ag2) ∉ Graph then Add Ag2 to Graph
                  Weight1(Arc(Ag1, Ag2)) := Str.TRUST(Ag2)Ag1
                  Weight2(Arc(Ag1, Ag2)) := Str.n(Ag2)Ag1
                  Potential_Witnesses := Potential_Witnesses ∪ {Ag2} } } } }
}
```

Algorithm 9.7

```
Evaluate-Node(Ag_y) {
  ∀Arc(Ag_x, Ag_y)
    If Node(Ag_x) is note evaluated Then
      Evaluate-Node(Ag_x)

  m1 := 0, m2 := 0
  ∀Arc(Ag_x, Ag_y) {
    m1 = m1 + Weight(Node(Ag_x)) * Weight(Arc(Ag_x, Ag_y))
    m2 = m2 + Weight(Node(Ag_x))
  }
  Weight(Node(Ag_y)) = m1 / m2
}
```

Algorithm 9.8

Algorithm 9.7: The construction of the trust graph is described as follows:

1- Agent Ag_a sends a request about the Ag_b's trustworthiness to all the confidence agents Ag_i. The nodes representing these agents (denoted *Node*(Ag_i)) are added to the graph. Since the trustworthiness values of these agents are known, the weights of these nodes (denoted *Weight*(*Node*(Ag_i))) can be evaluated. These weights are represented by $TRUST(Ag_i)_{Aga}$ (i.e by Ag_i's trustworthiness according to the point of view of Ag_a).

2- Ag_a uses the primitive *Send*(Ag_i, *Investigation*(Ag_b)) in order to ask Ag_i to offer a trustworthiness value for Ag_b. The Ag_is' answers are recovered when they are offered in a variable denoted *Str* by *Str* = *Receive*(Ag_i). *Str.Agents* represents the set of agents referred by Ag_i. $Str.TRUST(Ag_j)_{Agi}$ is the trustworthiness value of an agent Ag_j (belonging to the set *Str.Agents*) from the point of view of the agent which referred it (i,e, Ag_i).

3- When a consulted agent answers by indicating a set of agents, these agents will also be consulted. They can be regarded as potential witnesses. These witnesses are added to a set called: *Potonial_Witnesses*. When a potential witness is consulted, it is removed from the set.

4- To ensure that the evaluation process terminates, two limits are used: the maximum number of agents to be consulted (*Limit_Nbr_Visited_Agents*) and the maximum number of witnesses who must offer an answer (*Limit_Nbr_Witnesses*). The variable *Nbr_Additional_Agents* is used to be sure that the first limit is respected when Ag_a starts to receive the answers of the consulted agents.

Algorithm 9.8: The evaluation of a graph node is based on the trustworthiness combination formula (Equation 9.5). The weight of each node that represents the trustworthiness value of the agent represented by the node is evaluated on the basis of the weights of the adjacent nodes. For example, let *Arc*(Ag_x, Ag_y) an arc in the graph, before evaluating Ag_y it is necessary to evaluate Ag_x. Consequently, the evaluation algorithm is a recursive one. The algorithm terminates because the nodes of the set *Confidence*(Ag_a) are already evaluated by Algorithm 9.7. Since the evaluation is done recursively, the call of this algorithm in the main program has as parameter the agent Ag_b.

Complexity. Our trustworthiness model is based on the construction of a trust graph and on a recursive call to the function *Evaluate-Node*(Ag_y) to assess the weight of all the nodes. Since each node is visited exactly once, there are n recursive calls, where n is the number of nodes in the graph. To assess the weight of a node we need the weights of its neighboring nodes and the weights of the input edges. Thus, the algorithm takes a time in $O(n)$ for the recursive calls and a time in $O(a)$ to assess the agents' trustworthiness where a is the number of edges. The run time of the trustworthiness algorithm is therefore in $O(max(a, n))$ i.e. linear in the size of the graph.

In total, Algorithm 9.1 of our persuasion dialogue game protocol takes a time in:
$O(max(|S_{Ag1}|, |S_{Ag2}|) + max(a, n)) = O(max(|S_{Ag1}|, |S_{Ag2}|, a, n))$.

9.4 Implementation

In this section we describe the implementation of our persuasion dialogue game protocol (the different dialogue games and the trustworthiness model) using the Jack™ platform (The Agent Oriented Software Group, 2004). We chose this language for three main reasons:

1- It is an agent-oriented language offering a framework for multi-agent system development. This framework can support different agent models.
2- It is built on top of and fully integrated with the Java programming language. It includes all components of Java and it offers specific extensions to implement agents' behaviors.
3- It supports *logical variables* and *cursors*. A cursor is a representation of the results of a query. It is an enumerator which provides query result enumeration by means of re-binding the logical variables used in the query. These features are particularly helpful when querying the state of an agent's beliefs. Their semantics is mid-way between logic programming languages with the addition of type checking Java style and embedded SQL.

9.4.1 General Architecture

Our system consists of two types of agents: *conversational agents* and *trust model agents*. These agents are implemented as Jack™ agents, i.e. they inherit from the basic class Jack™ *Agent*. Conversational agents are agents that take part in the persuasion protocol. Trust model agents are agents that can inform an agent about the trustworthiness of another agent (Figure 9.5).

According to the specification of the Justification game (Section 9.2.4 (D)), an agent Ag_2 can play an acceptance or a refusal move according to whether it considers that its interlocutor Ag_1 is trustworthy or not. If Ag_1 is unknown for Ag_2, Ag_2 can ask agents that it considers trustworthy for it to offer a trustworthiness assessment of Ag_1. From the received answers, Ag_2 builds a trust graph and assesses the Ag_1's trustworthiness as explained in Section 9.3.3.

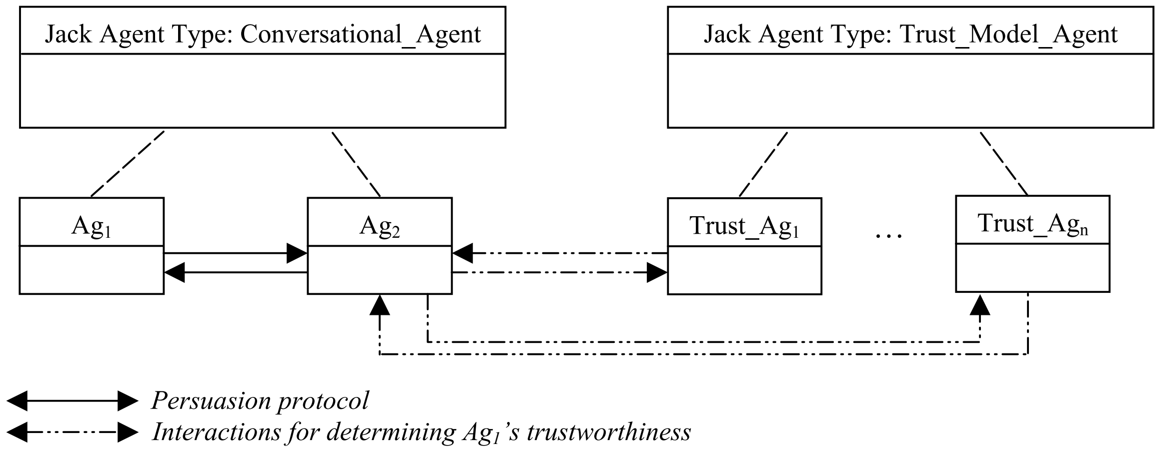


Figure 9.5. The general architecture of the system

To take part in our persuasion protocol, agents must have knowledge and argumentation systems. Agents' knowledge are implemented using Jack™ data structures called *beliefsets*. The argumentation systems are implemented as Java modules using a logical programming paradigm. These modules use agents' beliefsets to build arguments for or against certain propositional formulae. The actions that agents perform on commitments or on their contents are programmed as *events*. When an agent receives such an event, it seeks a *plan* to handle it. These plans are the algorithms 9.2, 9.3, 9.4, 9.5, and 9.6 presented in this chapter.

The trustworthiness model is implemented using the same principle (events + plans). The requests sent by an agent about the trustworthiness of another agent are events and the evaluations of agents' trustworthiness are programmed in plans. The trust graph is implemented as a Java data structure (oriented graph).

As Java classes, conversational agents and trust model agents have private data called *Belief Data*. For example, the different commitments and arguments that are created and manipulated are given by a data structure called *CAN* implemented using tables and the different actions expected by an agent in the context of a particular game are given by a data structure (table) called *data_expected_actions*. The different agents' trustworthiness values that an agent has are recorded in a data structure (table) called *data_trust*. These data and their types are given in Figures 9.6 and 9.7.

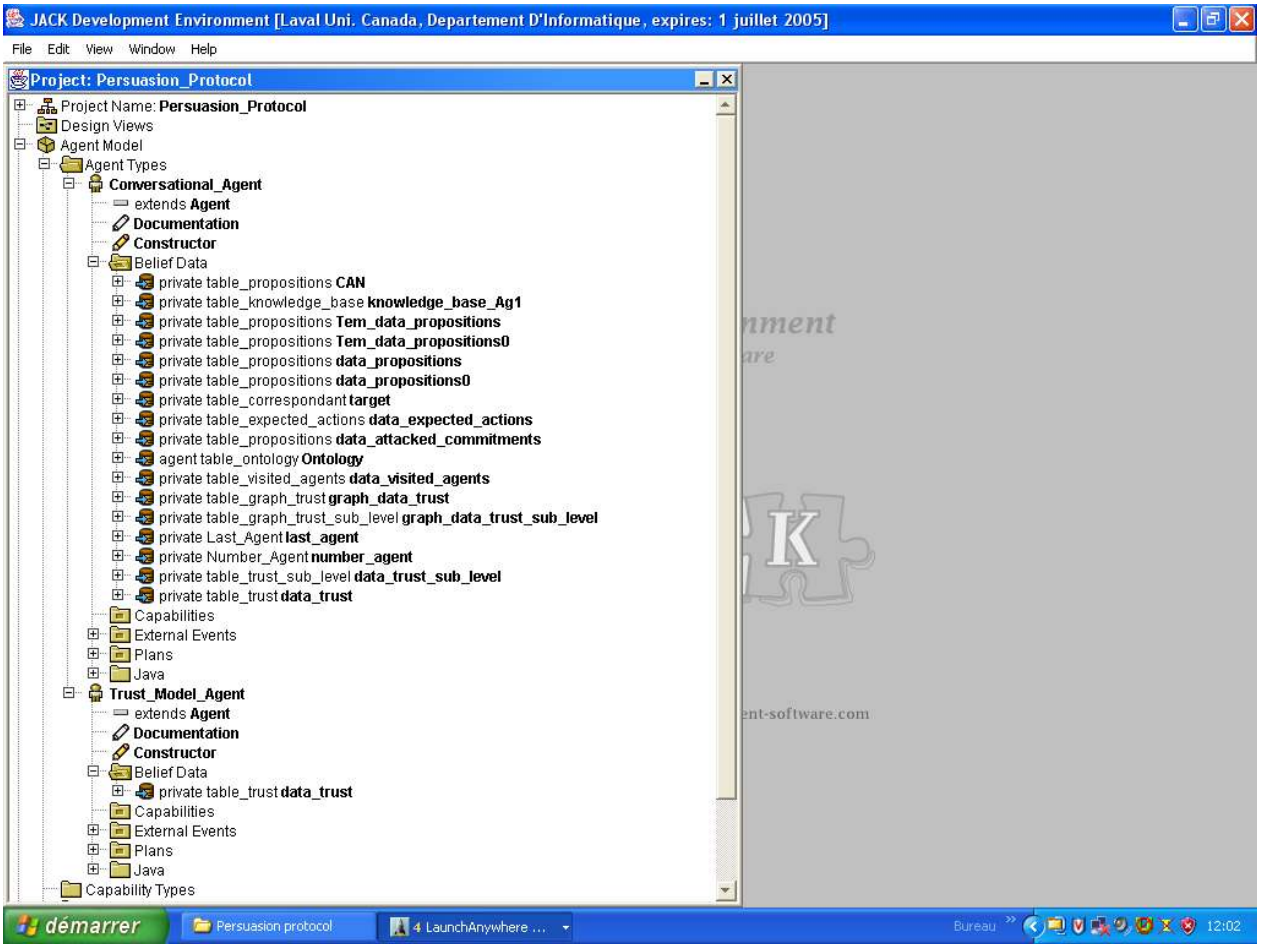


Figure 9.6. Belief Data used in our prototype

9.4.2 Implementation of the Trustworthiness Model

The trustworthiness model is implemented by agents of type: trust model agent. Each agent of this type has a knowledge base implemented using Jack^{TM} *beliefsets*. This knowledge base called *table_trust* has the following structure: *Agent_name*, *Agent_trust*, and *Interaction_number*. Thus, each agent has information on other agents about their trustworthiness and the number of times that it interacted with them. The visited agents during the evaluation process and the agents added in the trust graph are recorded in two Jack^{TM} *beliefsets* called: *table_visited_agents* and *table_graph_trust*. The two limits used in Algorithm 9.7 (*Limit_Nbr_Visited_Agents* and *Limit_Nbr_Witnesses*) and the trustworthiness threshold w are passed as parameters to the Jack^{TM} constructor of the original agent Ag_a that seeks to know if its interlocutor Ag_b is trustworthy or not. This original agent is a conversational agent.

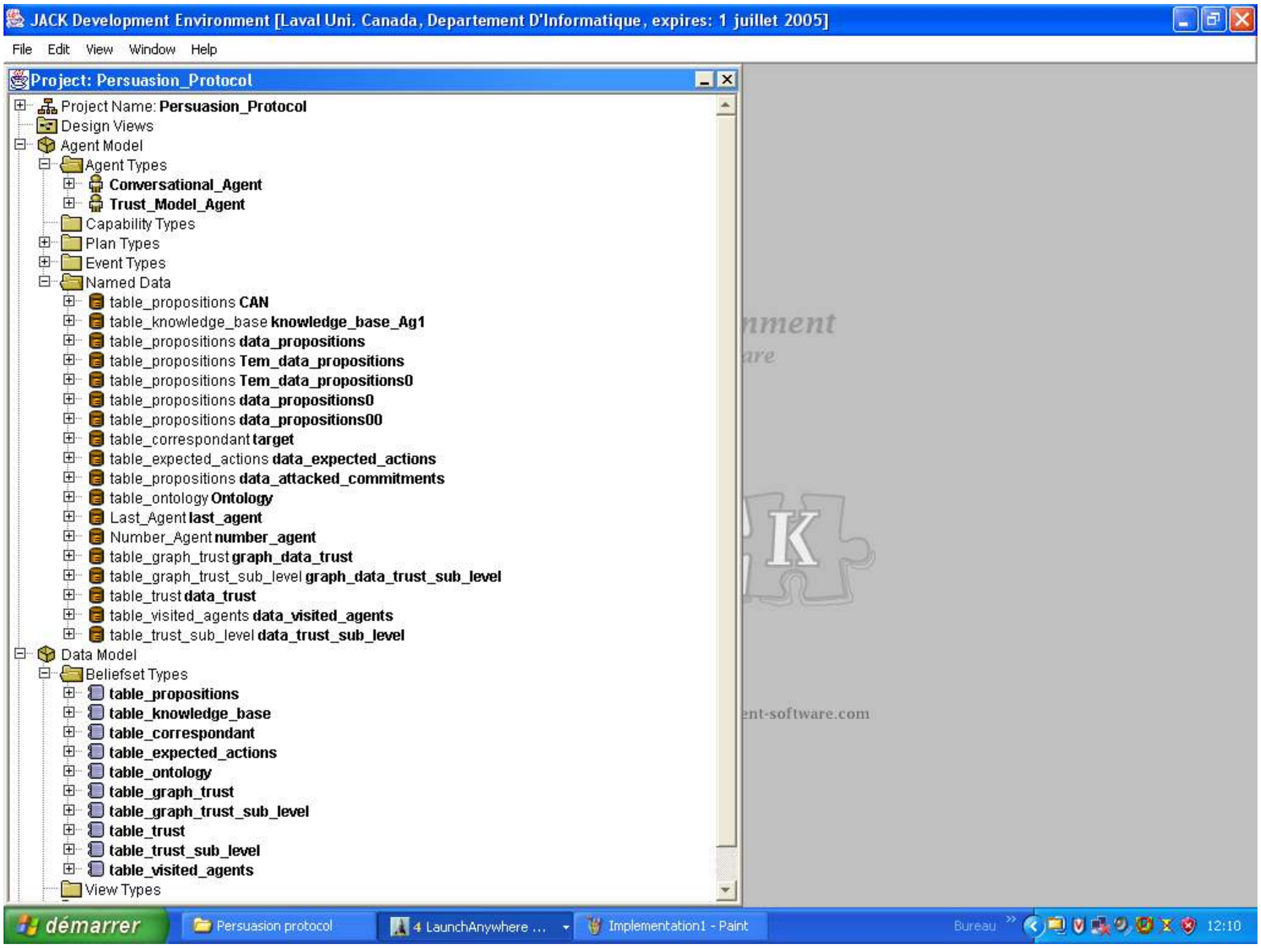


Figure 9.7. Beliefsets used in our prototype

The main steps of the evaluation process of Ag_b's trustworthiness are implemented as follows:

1- By respecting the two limits and the threshold w , Ag_a consults its knowledge base *data_trust* of type *table_trust* and sends a request to its confidence agents Ag_i (i = 1,.., n) about Ag_b's trustworthiness. The Jack™ primitive *Send* makes it possible to send the request as a Jack™ message that we call *Ask_Trust* of *MessageEvent* type. Ag_a sends this request starting by confidence agents whose trustworthiness value is highest.

2- In order to answer to the Ag_a's request, each agent Ag_i executes a Jack™ plan instance that we call *Plan_ev_Ask_Trust*. Thus, each agent Ag_i consults its knowledge base and offers to Ag_a an Ag_b's trustworthiness value if Ag_b is known by Ag_i. If not, Ag_i proposes a set of confidence agents from its point of view, with their trustworthiness values and the number of times that it interacted with them. In the first case, Ag_i sends to Ag_a a Jack™ message that we call *Trust_Value*. In the second case, Ag_i sends a message that we call *Confidence_Agent*. These two messages are of type *MessageEvent*.

3- When Ag_a receives the *Trust_Value* message, it executes a plan: *Plan_ev_Trust_Value*. According to this plan, Ag_a adds to a graph structure called *graph_data_trust* two information: 1) the agent Ag_i and its trustworthiness value as graph node, 2) The trustworthiness value that Ag_i offers for Ag_b and the number of times that Ag_i interacted with Ag_b as arc relating the node Ag_i and the node Ag_b. This first part of the trust graph is recorded until the end of the evaluation process of Ag_b's trustworthiness. When Ag_a receives the *Confidence_Agent* message, it executes another plan: *Plan_ev_Confidence_Agent*. According to this plan, Ag_a adds to another graph structure: *graph_data_trust_sub_level* three information for each Ag_i agent: 1) the agent Ag_i and its trustworthiness value as a sub-graph node, 2) the nodes Ag_j representing the agents proposed by Ag_i, 3) For each agent Ag_j, the trustworthiness value that Ag_i assigns to Ag_j and the number of times that Ag_i interacted with Ag_j as arc between Ag_i and Ag_j. This information that constitutes a sub-graph of the trust graph will be used to evaluate Ag_j's trustworthiness values using Equation 9.5. These values are recorded in a new structure: *new_data_trust*. Thus, the structure *graph_data_trust_sub_level* releases the memory once Ag_j's trustworthiness values are evaluated. This technique allows us to decrease the space complexity of our algorithm.

4- Steps 1, 2 and 3 are applied again by substituting *data_trust* by *new_data_trust*, until all the consulted agents offer a trustworthiness value for Ag_b or until one of the two limits (*Limit_Nbr_Visited_Agents* or *Limit_Nbr_Witnesses*) is reached.

5- Evaluate the Ag_b's trustworthiness value using the information recorded in the structure *graph_data_trust* by applying Equation 9.5.

The different events and plans implementing our trustworthiness model and the conversational agent constructor are illustrated by Figure 9.8. Figure 9.9 illustrates an example generated by our prototype of the process allowing an agent Ag_1 to assess the trustworthiness of another agent Ag_2 in a domain related to the example given in Section 9.4.3. In this example, Ag_2 is considered trustworthy by Ag_1 because its trustworthiness value (0.79) is higher than the threshold (0.7).

9.4.3 Implementation of the Dialogue Games

In our system, agents' knowledge bases contain propositional formulae and arguments. These knowledge bases are implemented as Jack™ *beliefsets*. *Beliefsets* are used to maintain an agent's beliefs about the world. These beliefs are represented in a first order logic and tuple-based relational model. The logical consistency of the beliefs contained in a *beliefset* is automatically maintained. The advantage of using *beliefsets* over normal Java data structures is that *beliefsets* have been specifically designed to work within the agent-oriented paradigm.

Our knowledge bases (KBs) contain two types of information: arguments and beliefs. Arguments have the form ([*Support*], *Conclusion*), where *Support* is a set of propositional formulae and *Conclusion* is a propositional formula. Beliefs have the form ([*Belief*], *Belief*) i.e. *Support* and *Conclusion* are identical. The meaning of the propositional formulae (i.e. the ontology) is recorded in a *beliefset* called *table_ontology* whose access is shared between the two agents. This beliefset has two fields: *Proposition* and *Meaning*.

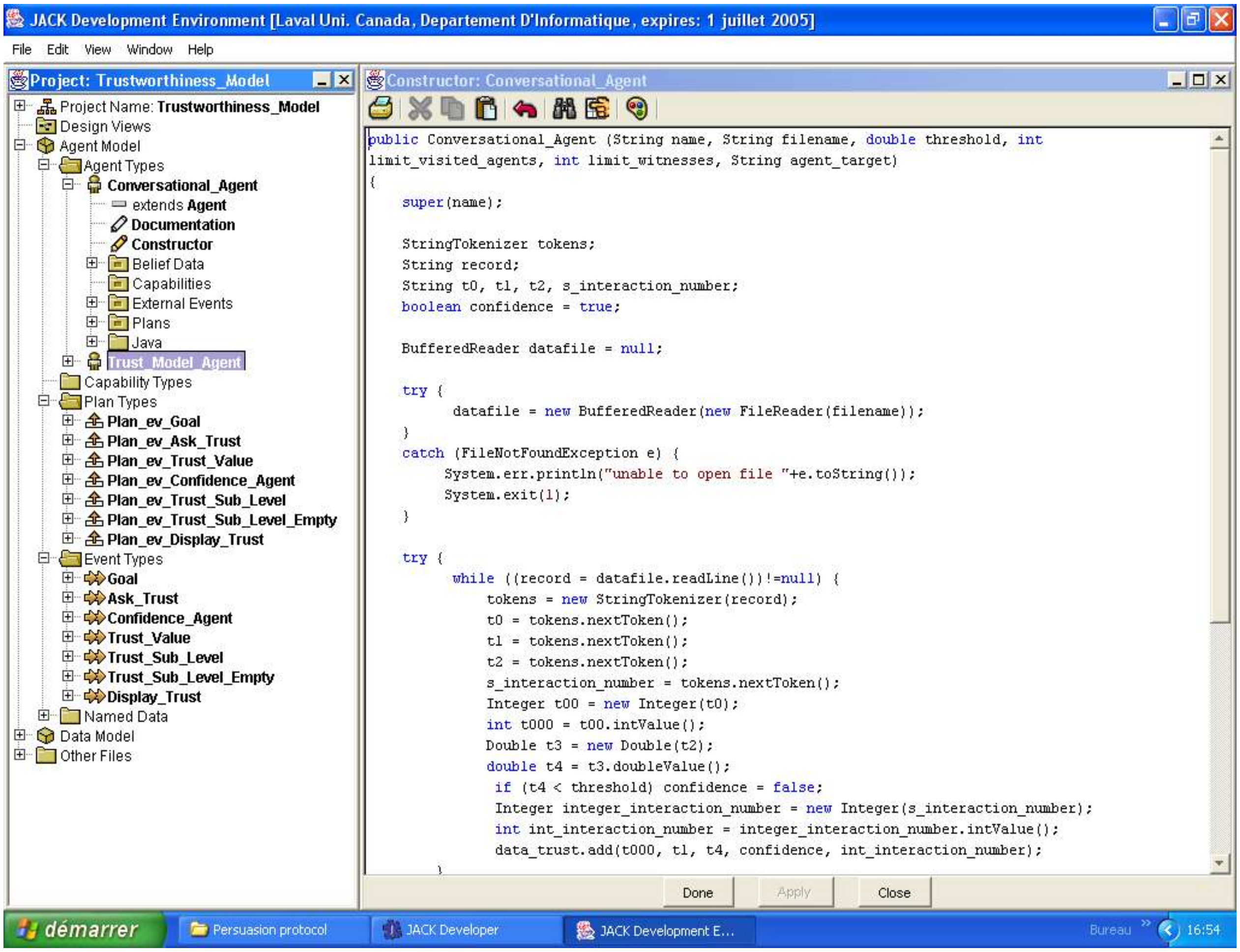


Figure 9.8. Events, plans and the conversational agent constructor implementing the trustworthiness model

To open a dialogue game, an agent uses its argumentation system. The argumentation system allows this agent to seek in its knowledge base an argument for a given conclusion or for its negation (*"against argument"*). For example, before creating a commitment $SC(Id_0, Ag_1, Ag_2, p)$, agent Ag_1 must find an argument for p. This enables us to respect the commitment semantics by making sure that agents can always defend the content of their commitments. The argumentation system of an agent is implemented using *logical statements*, *logical members* and *cursors*. Logical statements follow *Open World* semantics that models real world knowledge. It allows for three truth states: true, false and unknown. Logical members bring elements of logic programming to Jack^{TM}. They follow the semantic behavior of variables from logic programming languages such as prolog. That is, they are not place-holders for assigned values like normal Java variables. Rather, they represent a specific, but possibly unknown, value. Conclusions and supports of arguments are logical members, and statements using these conclusions and supports are logical statements. Cursors allow agents to seek an argument for supporting a given conclusion, using the query method of a knowledge base.

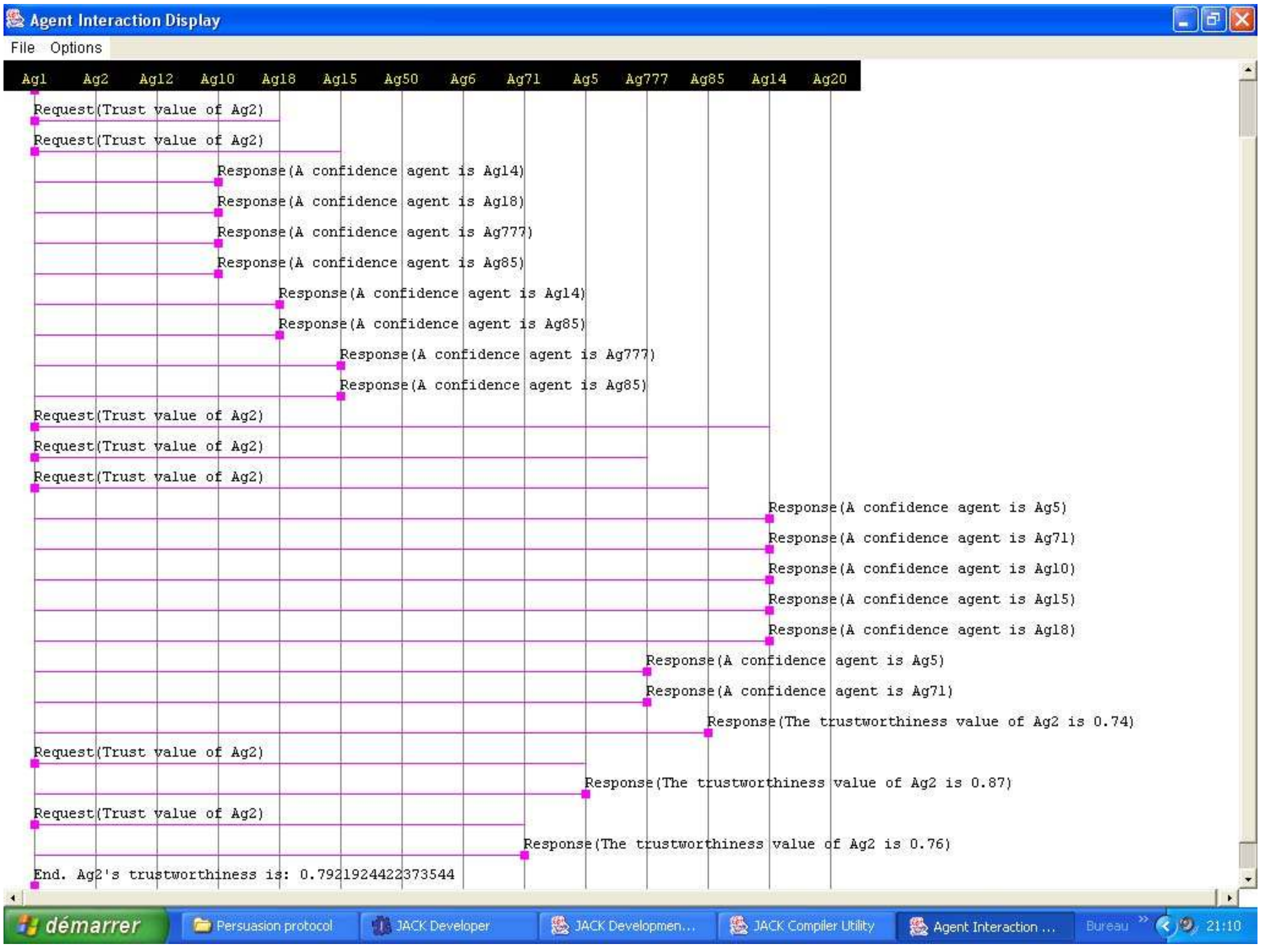


Figure 9.9. The screen shot of a trustworthiness evaluation process

Agent communication is done by sending and receiving messages. These messages are *events* that extend the basic Jack™ *event*: *MessageEvent* class. *MessageEvents* represent events that are used to communicate with other agents. Whenever an agent needs to send a message to another agent, this information is packaged and sent as a *MessageEvent*. A *MessageEvent* can be sent using the primitive: *Send*(*Destination*, *Message*). In our protocol, *Message* represents the action that an agent applies to a commitment or to its content, for example: $Create(Ag_1, SC(Id_0, Ag_1, Ag_2, p))$, etc.

Our dialogue games are implemented as a set of *events* (*MessageEvents*) and *plans*. A plan describes a sequence of actions that an agent can perform when an event occurs. Whenever an event is posted and an agent chooses a task to handle it, the first thing the agent does is to try to find a plan to handle the event. Plans are reasoning methods describing what an agent should do when a given event occurs.

Each dialogue game corresponds to an event and a plan. These games are not implemented within the agents' program, but as event classes and plan classes that are external to agents. Thus, each conversational agent can instantiate these classes. An agent Ag_1 starts a dialogue game by generating an event and by sending it to its interlocutor *Ag_2*. *Ag_2* executes the plan corresponding to the received event and answers by generating another event and by sending it to *Ag_1*. Consequently, the two agents can communicate by using the same protocol since they can instantiate the same classes representing the events and the plans. For example, the event *Event_Attack_Commitment* and the plan *Plan_ev_Attack_commitment* implement the defense game. The architecture of our conversational agents is illustrated in Figure 9.10.The different events and plans implementing our dialogue games are given in Figure 9.11. Figure 9.12 illustrates the screen shot of the example presented in Section 9.2.5.

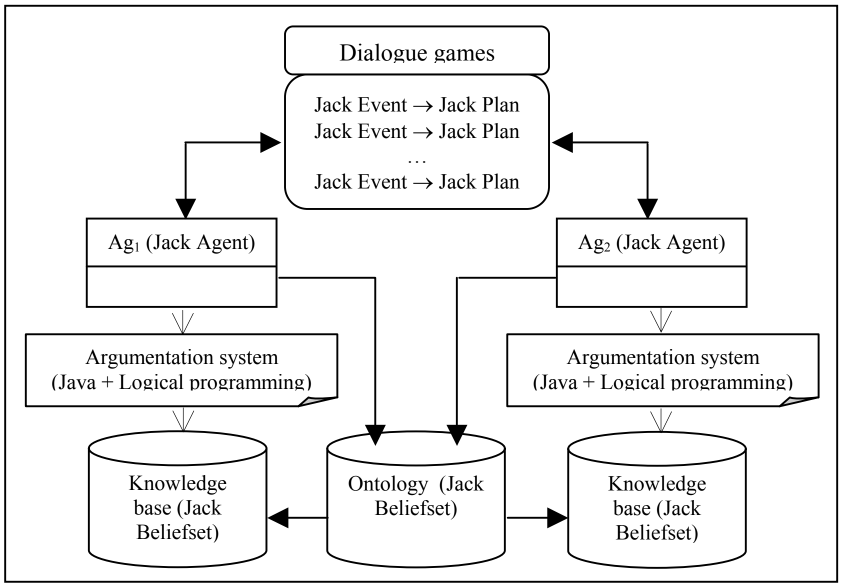


Figure 9.10. The architecture of the conversational agents

To start the entry game, an agent (initiator) chooses a goal that it tries to achieve. This goal is to persuade its interlocutor that a given propositional formula is true. For this reason, we use a particular event: *BDI Event* (*Belief-Desire-Intention*). BDI events model goal-directed behavior in agents, rather than plan-directed behavior. What is important is the desired outcome, not the method chosen to achieve it. This type of events allows an agent to pursue long term goals.

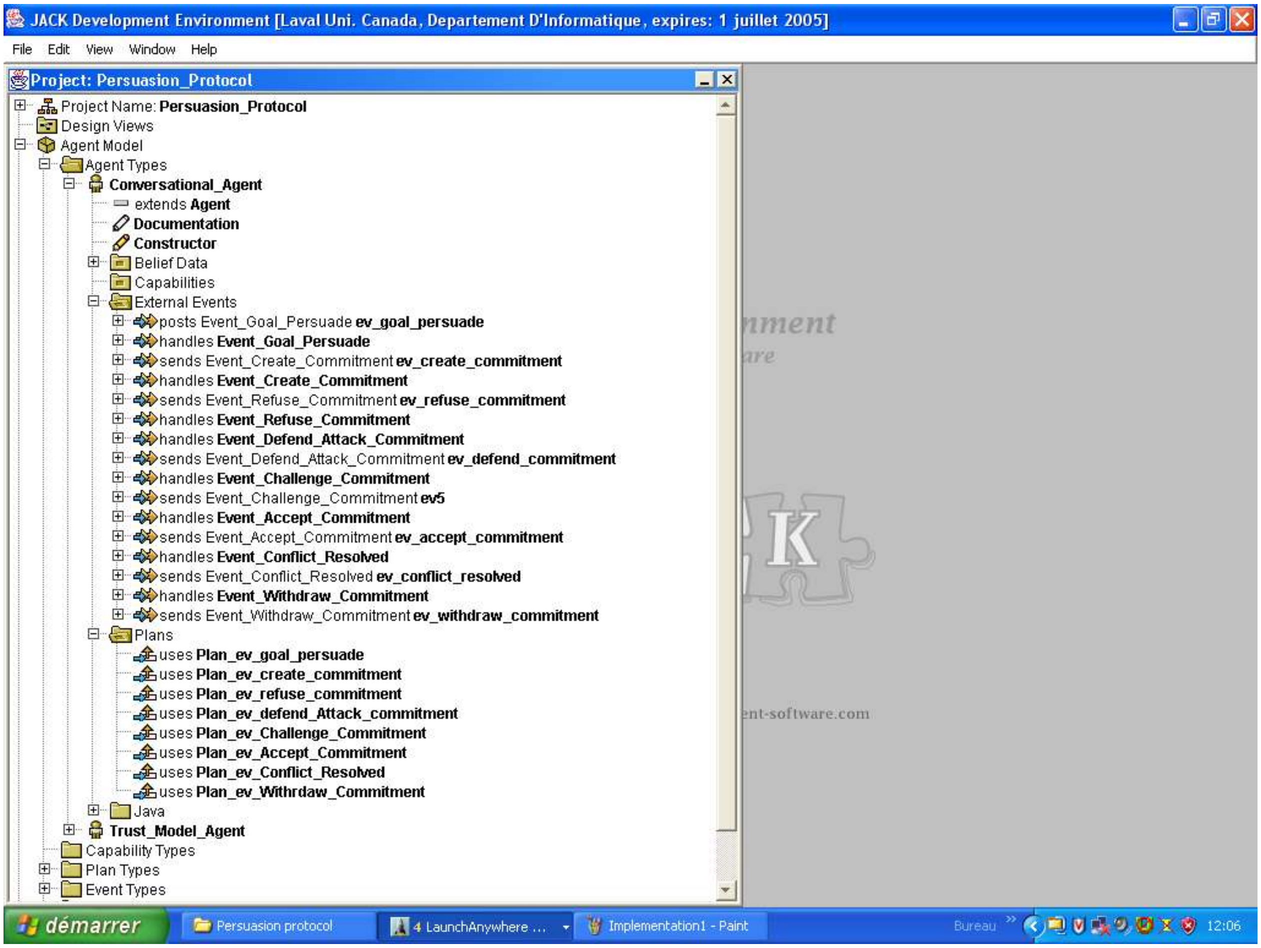


Figure 9.11. Events and plans implementing the dialogue games

9.5 Related Work

In this section, we compare our protocol with some proposals that have been put forward in two domains: dialogue modeling and commitment based protocols.

1- ***Dialogue modeling***. In (Amgoud et al., 2000a, 2000b) and (Parsons et al., 2003) Amgoud, Parsons and their colleagues studied argumentation-based dialogues. They proposed a set of atomic protocols which can be combined. These protocols are described as a set of dialogue moves using Walton and Krabbe's classification and formal dialectics. In these protocols, agents can argue about the truth of propositions. Agents can communicate both propositional statements and arguments about these statements. These protocols have the advantage of taking into account the capacity of agents to reason as well as their attitudes (confident, careful, etc.). In addition, Prakken (2001) proposed a framework for protocols for dynamic disputes, i.e., disputes in which the available information can change during the conversation. This framework is based on a logic of defeasible argumentation and is formulated for dialectical proof theories. Soundness and completeness of these protocols have also been studied. In the same direction, Brewka (2001) developed a formal model for argumentation processes that combines nonmonotonic logic with protocols for dispute. Brewka pays more attention to the speech act aspects of disputes and he formalizes dispositional protocols in situation calculus. Such a logical formalization of protocols allows him to define protocols in which the legality of a move can be disputed. Semantically, Amgoud, Parsons, Prakken and Brewkas' approaches use a defeasible logic. Therefore, it is difficult, if not impossible, to formally verify the proposed protocols.

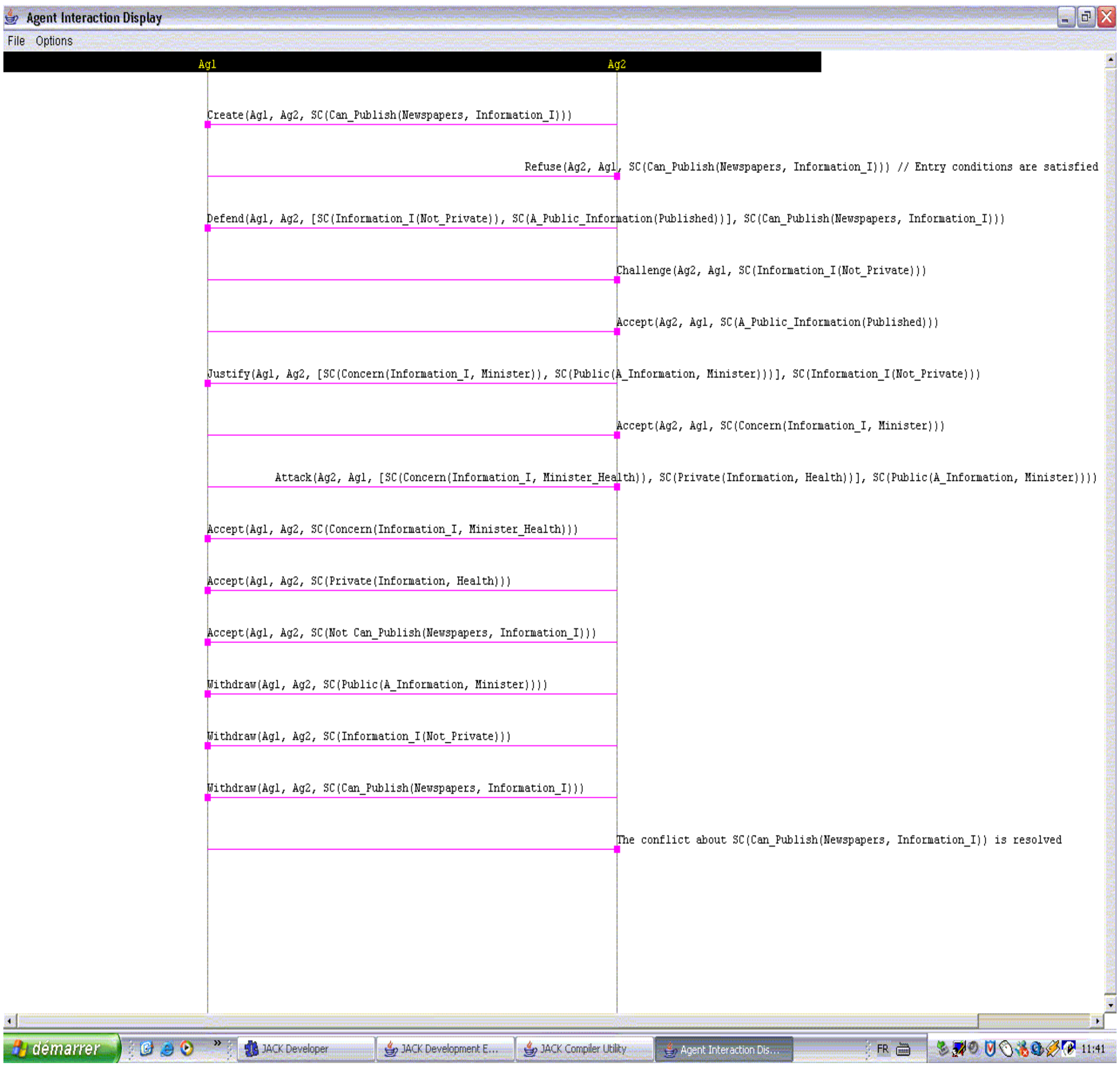


Figure 9.12. The example screen shot

There are many differences between our protocol and the protocols proposed in the domain of dialogue modeling:

1. Our protocol uses not only an argumentative approach, but also a public one. The effects of utterances are formalized not in terms of agents' private attitudes (beliefs, intentions,

etc.), but in terms of social commitments. In opposition of private mental attitudes, social commitments can be verified.

2. Our protocol is based on a combination of dialogue games instead of simple dialogue moves. Using our dialogue game specifications enables us to specify the entry and the exit conditions more clearly. In addition, computationally speaking, dialogue games provide a good balance between large protocols that are very rigid and atomic protocols that are very detailed.

3. From a theoretical point of view, Amgoud, Parsons, Prakken and Brewkas' protocols use moves from formal dialectics, whereas our protocol uses actions that agents apply on commitments. These actions capture the speech acts that agents perform when conversing. The advantage of using these actions is that they enable us to better represent the persuasion dynamics considering that their semantics is defined in an unambiguous way in a temporal and dynamic logic (see Chapter 7). Specifying protocols in this logic allows us to formally verify these protocols using model checking techniques (see Chapter 8).

4. Amgoud, Parsons and Prakkens' protocols use only assertion, acceptance, refusal and challenge moves, whereas our protocol uses not only creation, acceptance, refusal and challenge actions, but also justify, attack and defense actions in an explicit way. These argumentation relations allow us to directly illustrate the concept of dispute in this type of protocols.

5. Amgoud, Parsons, Prakken and Brewka use an acceptance criterion directly related to the argumentation system, whereas we use an acceptance criteria for conversational agents (supports of arguments and trustworthiness). This makes it possible to decrease the computational complexity of the protocol for agent communication. The reason is that in the approach proposed by Amgoud, Parsons, Prakken and Brewka, to decide about the acceptance of each argument, we need to find a least fixpoint of a given function. This task is computationally complex. In addition, in the literature there is no implementation of argumentative-based protocols.

2- Commitment-based protocols. Yolum and Singh (2002) developed an approach for specifying protocols in which actions' content is captured through agents' commitments. They provide operations and reasoning rules to capture the evolution of commitments. In a similar way, Fornara and Colombetti (2003) proposed a method to define interaction protocols. This method is based on the specification of an interaction diagram (ID) specifying which actions can be performed under given conditions. These approaches allow them to represent the interaction dynamics through the allowed operations. Our protocol is comparable to these protocols because it is also based on commitments. However, it is different in the following respects. The choice of the various operations is explicitly dealt with in our protocol by using argumentation and trustworthiness. In commitment-based protocols, there is no indication about the combination of different protocols. However, this notion is essential in our protocol using dialogue games. Unlike commitment-based protocols, our protocol plays the role of the dialectical proof theory of an argumentation system. This enables us to represent different dialogue types as studied in the philosophy of language. Finally, we provide a termination proof of our protocol and a complexity analysis of our implementation whereas these properties are not yet studied in classical commitment-based protocols.

9.6 Discussion

The protocol that we proposed in this chapter is more flexible than the traditional protocols of agent communication for the following reasons:

1- Our protocol is not specified in a static way, but results from the combination of different dialogue games. How these dialogue games can be combined is not fixed in advance, but depends on the evolution of the communication. Consequently, the protocol automaton is non-deterministic.

2- Agents can reason about the protocol using their argumentation systems and the trustworthiness model. The agents' choices depend on the current state of the dialogue in terms of the states of the different commitments and arguments (i.e. the current state of the CAN). Therefore, which games agents can play are determined on the fly.

3- Our protocol specifies the combination rules of different dialogue games and how agents can use these rules in a logical way. An interesting consequence of this specification is that the protocol does not have the problem of managing exceptions (messages not specified by the protocol). The reason is that the protocol does not specify a fixed number of possibilities, but only the logical rules that agents can use and reason about in any situations.

9.7 Conclusion

The contribution of this chapter is the proposition of a logical language for specifying persuasion protocols between autonomous agents using our commitment and argument approach. This language has the advantage of expressing the public elements and the reasoning process that allows agents to choose an action among several possible actions. Because our protocol is defined as a set of dialogue games, this protocol is more flexible than the traditional protocols such as those used in FIPA-ACL. This flexibility results from the fact that these games can be combined to produce complete and more complex protocols and from the fact that agents can reason about the protocol. We formalized these games as a set of conversation policies, and we described the persuasion dynamics by the combination of five dialogue games. Another contribution of this chapter is the tableau-based termination proof of the protocol. We also implemented this protocol using an agent-oriented language and a logical programming paradigm and we analyzed its computational complexity. Finally, we presented an example to illustrate the persuasion dynamics by the combination of different dialogue games.

Chapter 10

Conclusion

10.1 General Discussion

In this thesis we proposed a unified framework for the pragmatics and the semantics of agent communication. Our framework has the advantage of being based on solid philosophical foundations and equipped with a logical formalization. The philosophical foundations are supplied by the philosophical definition of social commitments, Speech Act Theory and formal dialectics (the philosophy of arguments). The logical formalization is defined in terms of a combination of branching time logic (CTL*) and dynamic logic.

Another advantage of this framework lies in the fact that it captures both the pragmatics and semantics of agent interactions. We discuss these two aspects in this section.

Pragmatics: The interactions between autonomous agents are reflected by the actions that they perform on commitments and on their contents. These actions can be supported by arguments. The dynamics of the interactions is reflected by the creation of commitments, by the agents' positioning on these commitments (acceptance, refusal, challenge, attack, etc.), and by the evolution of commitment states in time (satisfied, withdrawn, etc). All the commitments and arguments handled in an interaction can be represented using our commitment and argument networks (CAN). This formalism allows us to model the dynamics of conversations and offers an external representation of the conversational activity. This notion of external representation is very useful because it provides participants with a common understanding of the current state of the conversation and its advancement. The formalism also allows us to ensure conversational consistency when considering the actions performed by the agents. It relies on our approach combining commitments and arguments. This approach has the advantage of capturing both the social and public aspects of a conversation, and the reasoning aspect required in order to take part in conversations. Thus, the formalism can clearly illustrate the creation phases of new commitments and the positioning phases on these commitments, as well as the argumentation and justification phases.

Semantics: All the elements captured by the pragmatic aspects of our framework are semantically defined in a logical formalism combining temporal and dynamic logics ($DCTL*_{CAN}$). The concept of social commitment, the different types of commitments and the concept of argument are defined as modal operators logic. The actions that agents apply to commitments and on their contents as well as the argumentation relations are defined using the *Perform* operator that reflects the performance of actions. The important link between commitments and arguments that we established in the pragmatic level is formally captured by the semantics in the form of properties using the other elements of the logical model. Our semantics offers a clear and unambiguous means to introduce the different elements and the various operations that we described in the pragmatic level of agent communication. It can also be used for verification purposes. A direct application is to check if a particular protocol (for example a negotiation or a persuasion protocol) respects the introduced specifications.

Our pragmatic approach presented in Chapter 5 is different from the social approach proposed by Singh (1998, 2000) and Colombetti (2000) in the sense that social commitments in our approach are not only public states but also deontic notions. Agents must justify and defend their commitments if necessary. Thanks to the link we established between commitments and arguments, agents can reason about their commitments and consequently can communicate in a flexible way. In addition, there are many differences between our approach and the argumentative approach proposed by Amgoud and her colleagues (2002a, 2000b). The main difference is that Amgoud et al.'s proposal is based upon dialectical systems, and the evolution of agent conversations is captured using the commitment stores that only record what is uttered during the conversation (MacKenzie, 1979). However, in our approach, the evolution is captured by the notion of commitment and commitment content states that evolve as a result of the actions that agents perform when conversing (creation, withdrawal, reactivation, violation and satisfaction). The main idea of our approach is that agent communication is considered as actions that agents perform on social commitments and arguments. Thus, different speech act types can be expressed using these actions.

The CAN formalism presented in Chapter 6 as the basis of our pragmatic approach allows us to represent the dynamics of agent communication in a formal way. This new formalism for agent communication is different from all other agent communication formalisms proposed in (Pitt and Mamdani, 2000), (FIPA-ACL, 2001), (Yolum and Singh, 2002) and (Fornara and Colombetti, 2003). Unlike these formalisms, the CAN formalism can be used as a means to help agents to participate in conversations. In addition, this formalism enables agents to reason about their communicative acts and about the current state of the conversation in order to decide about the next actions to be performed. This reasoning aspect is tied to the agents' argumentation systems.

Semantically speaking, our logical model presented in Chapter 7 is different from the semantics defined by Singh (2000) and by Verdicchio and Colombetti (2003). Our semantics is based not only on a temporal logic, but also on a dynamic logic and it captures different commitment types, different commitment states and different actions performed on commitments. Our semantics is defined as a model-theoretic semantics that can be successfully used to capture the semantics of defeasible arguments. It is therefore different from the semantics defined in (Amgoud et al., 2002) which is based on an informal logic. Another difference is that our semantic framework can be used to express the meaning of different speech act types.

In addition, in Chapter 8, we proposed a new model checking algorithm for the verification of dialogue game protocols whose complexity matches that of the best existing algorithms.

Our model-checking technique allows us not only to verify if the dialogue game protocol satisfies a given property expressed in our DCTL*$_{CAN}$, but also if this protocol respects a simplified version of the tableau semantics of the communicative acts. To our knowledge, this model-checking technique is the first proposal in the domain of dialogue game verification.

Finally, there are many differences between our dialogue game protocol presented in Chapter 9 and the other dialogue game protocols discussed in Chapter 3. The main differences are:
1- Our proposal is based on a social and argument approach. Consequently, agents can reason about their actions in order to decide about the dialogue game to be played.
2- The decision making process is based not only on the agents' argumentation systems, but also on the agents' trustworthiness.
In addition, we provided a termination proof of our protocol, and we discussed its computational complexity.

10.2 Contributions

The main contributions of this thesis are:

1- A formal pragmatic approach capturing the conversations' public elements and the agents' reasoning mechanisms using their private states for modeling agent communication. This approach was published in (Bentahar et al., 2003).

2- A formalism called Commitment and Argument Network representing the dynamics of agent communication and helping agents to participate in conversations in a flexible way. This main contribution resulted in two publications: (Bentahar et al., 2004b, 2004c). Together, contributions 1 and 2 allowed us to achieve our first and second objectives stated in Chapter 1.

3- A model-theoretic semantics for the pragmatic approach defining the meaning of the different communicative acts that we use in our pragmatic approach, especially the ones commonly used in multi-agent interactions, and capturing the semantics of defeasible arguments. This semantics resulted in two publications (Bentahar et al., 2004e, 2004f). This contribution matches the third objective of this thesis.

4- A tableau-based model checking technique for the verification of dialogue game protocols specified in our framework. This contribution is published in an internal report (Bentahar and Moulin, 2004), and it is the subject of a submitted paper (Bentahar et al., 2005). This verification method is the fourth objective that we set in Chapter 1.

5- A new persuasion dialogue game protocol specified in our framework using a logical language, and implemented using an agent-oriented programming language. This contribution that matches the fifth objective of this thesis is published in (Bentahar et al., 2004d). The algorithmic specification of this protocol in the context of the CAN framework was the subject of another publication (Bentahar et al., 2004a).

Thus, all the objectives of this thesis are reached. In addition, contributions 1, 2, and 5 answer the first research question stated in Chapter 1: "*How may autonomous agents participate in conversations in a flexible way?*" Contribution 3 answers the second research question: "*How can we unify pragmatic and semantic approaches and how can the link between pragmatics and semantics be established in such an approach?*" Finally, contributions 4 and 5 answer the third research question: "*How can we formally specify and verify the agent communication mechanisms?*"

10.3 Future Work

As future work we intend:

1- To use our unified framework to specify other sophisticated protocols according to Walton and Krabbe's classification. Because this framework is based on a commitment and argument approach, the dialogue types in this dialectical-based classification can be supported. An important result of this work is to explain and formalize the shift between these different dialogue types during a conversation. The idea is to define a general dialogue-game protocol combining the different protocols (the combined protocol). The rules defining the dialectical shifts can be expressed in a logical language extending the one we proposed in Chapter 8. The implementation of such a protocol can be done using the same logic-programming and agent-oriented paradigm that we used for the persuasion dialogue game.

2- To define an operational and a denotational semantics for the different protocols and for the combined protocol. The operational semantics constitutes a means to formally derive the computation steps of the protocols. The denotational semantics provides a tool for specifying the compositionality of these protocols.

3- To implement and evaluate the model checking technique proposed in Chapter 8. The ABTA for $DCTL^*_{CAN}$ will be implemented in the Concurrency WorkBench of the New Century CWB-NC verification tool (Cleaveland and Sims, 1996). The ABTA manipulation procedure will be implemented in Standard ML. This work will be done in collaboration with Rance Cleaveland from State University of New York at Stony Brook.

4- To define a model checking technique for all the logic proposed in Chapter 7. The tableau technique proposed in Chapter 8 can be improved to support the complete version of the logic.

5- To explore other argumentation models, particularly Toulmin's model (1958) that is widely cited in the philosophy of argumentation, but still unexplored in the domain of agent communication.

Bibliography

Abdul-Rahman, A. and Hailes. S. Supporting trust in virtual communities. In Proc. Of the 33rd Hawaii Int. Conf. On Systems Science, 2000.

Adi, K., Debbabi, M., and Mejri, M. A new logic for electronic commerce protocols. In *Theoretical Computer Science*, vol. 291, 2003, pp. 223-283.

Allen, J.F., and Perrault, C.R. Analysing intention utterances. In *Artificial Intelligence*, vol. 15, 1980, pp. 143-178.

Alur, R., Henzinger, T.A., and Kupferman, O. Alternating time temporal logic. In Proc. Of the 38th IEEE Symp. On Foundations of Computer Science, 1997, pp. 100-109.

Amgoud, L. and Parsons, S. Agent dialogues with conflicting preferences. In Proc. of the 8th Int. Workshop on Agent Theories, Architectures, and Languages, Meyer, J-J. Ch. and Tambe, M. (eds.), 2001, pp. 1-14.

Amgoud, L. Contribution à l'intégration des préférences dans le raisonnement argumentatif. PhD. Thesis of Université Paul Sabatier-Toulouse, France, 1999.

Amgoud, L., and Maudet, N. Strategical considerations for argumentative agents. In Proc. of 10th Int. Workshop on Non-Monotonic Reasoning, 2002, pp. 409-417.

Amgoud, L., Maudet, N. and Parsons, S. An argumentation-based semantics for agent communication languages. In Proc. of 15th European Conf. on Artificial Intelligence, 2002, pp. 38-42.

Amgoud, L., Maudet, N., and Parsons, S. Modelling dialogues using argumentation. In Proc. of the 4th Int. Conf. On Multi-Agent Systems, 2000a, pp. 31-38.

Amgoud, L., Parsons, S. and Maudet, N. Arguments, dialogue, and negotiation. In Proc. of the 14th European Conf. On Artificial Intelligence, 2000b, pp. 338-342.

Aristotle. *Rhetorica*. 350 B.C. Tr. Freese, J.H. Leob, London, 1926.

Aristotle. *Topics*. 350 B.C. Clarendon Press, Oxford, UK, 1928. (W.D. Ross, Editor).

Austin, J.L. How to do things with words. Oxford University Press, England, 1962.

Baldoni, M., Baroglio, C., Martelli, A., Patti, V., and Schifanella, C. Verifying protocol conformance for logic-based communicating agents. In Proc. of the 5th Int. Workshop on Computational Logic in Multi-Agent Systems, 2004, pp. 82-97.

Barbuceanu, M. and Fox, M. COOL: a language for describing coordination in multi-agent systems, In Proc. of the 1st Int. Conf. on Multi-Agent Systems, 1995, pp. 17-25.

Belnap, N. and Perloff, M. The way of the agent. In *Studia Logica*, Kluwer, vol. 51, 1992, pp. 463-484.

Belnap, N. Backwards and towards in the modal logic of Agency. In *Philosophy and Phenomenological Research*, vol. 51, 1991, pp. 777-807.

Ben-Ari, M., Pnueli, A. and Manna, Z. The temporal logic of branching time. In Proc. of 8th Annual Sym. on Principles of Programming Languages, pp. 164-176. Journal Version, *Acta Informatica*, vol. 20, 1983, pp. 207-226.

Bench-Capon, T.J.M., Freeman J.B., Hohmann, H., and Prakken, H. Computational models, argumentation theories and legal practice. In *Argumentation Machines. New Frontiers in Argument and Computation* Reed, C. and Norman, T.J. (eds.), Kluwer Argumentation Library, 2003, pp. 85-120.

Benerecetti, M. and Cimatti, A. Symbolic model checking for multi-agent systems. In proc of the Int. Workshop on Model Checking and Artificial Intelligence, 2002, pp. 1-8.

Bentahar, J. and Moulin, B. On the verification of dialogue game protocols for communicating agents (a model checking approach). Internal Rapport of Research, Université Laval, Faculté des Sciences et de Génie, DIUL-RR-0403, 2004, pp. 1-34.

Bentahar, J. Moulin, B., and Chaib-draa, B. Specifying and implementing a persuasion dialogue game using commitment and argument network. In Rahwan, I., Moraitis, P., and Reed, C. (eds.), Int. Workshop on Argumentation in Multi-Agent Sysytems, AAMAS'04, LNAI, Springer, 2004a (in press).

Bentahar, J., Moulin, B. and Chaib-draa, B. Commitment and argument network: A new formalism for agent communication. In *Advances in Agent Communication*, Dignum, F. (ed.), Int. Workshop on Agent Communication Languages, AAMAS'03, LNAI 2922, Springer, 2004b, pp. 146-165.

Bentahar, J., Moulin, B. and Chaib-draa, B. Vers une approche pour la modélisation du dialogue basée sur les engagements et les arguments. In Actes des Secondes Journées Francophones Modèles Formels de l'Interaction, 2003, pp. 19-28.

Bentahar, J., Moulin, B., and Chaib-draa, B. Commitment and argument network: a formal framework for representing conversation dynamics. In *Logic and Dialogue*, Caelen, J., Vanderveken, D. and Vernant, D. (eds.), the Netherlands, Dordrecht, Kluwer, 2004c (in press).

Bentahar, J., Moulin, B., Meyer, J-J.Ch. and Chaib-draa, B. A computational model for conversation policies for agent communication. In Proc. of the 5th Int. Workshop on Computational Logic in Multi-Agent Systems, 2004d, pp. 66-81.

Bentahar, J., Moulin, B., Meyer, J-J.Ch. and Chaib-draa, B. A logical model for commitment and argument network for agent communication. In Proc. of 3rd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems, ACM Press, 2004e, pp.19-23.

Bentahar, J., Moulin, B., Meyer, J-J.Ch. and Chaib-draa, B. A modal semantics for an argumentation-based pragmatics for agent communication. In Rahwan, I., Moraitis, P. and Reed, C. (eds.), LNAI, Springer, 2004f (in press).

Bernholtz, O., Vardi, M.Y., and Wolper, P. An automata-theoretic approach to branching-time model checking. In *Computer Aided Verification*, Dill, D.L. (ed.), LNCS 818, Springer, 1994, pp. 142-155.

Bhat, G. and Cleaveland, R. Efficient model checking via the equational μ-calculus. In the 11th Annual Sym. on Logic in Computer Science, IEEE Computer Society Press, 1996, pp. 304-312.

Bhat, G. Tableau-based approaches to model-checking. PhD Thesis of North Carolina State University, USA, 1998.

Bhat, G., Cleaveland, R., and Groce, A. Efficient model checking via Büchi tableau automata. In *Computer-Aided Verification* Berry, G., Comon, H. and Finkel, A. (eds.), LNCS 2102, Springer, 2001, pp. 38-52.

Bordini, R.H., Fisher, M., Pardavila, C. and Wooldridge, M. Model checking AgentSpeak. In Proc. of the 2nd Int. Joint Conf. On Autonomous Agents and Multi Agent Systems, 2003a, pp. 409-416.

Bordini, R.H., Visser, W., Fisher, M., Pardavila, C., and Wooldridge, M. Model checking multi-agent programs with CASP. In *Computer-Aided Verification*, Hunt, W.A. and Somenzi, F. (eds.), LNCS 2725, Springer, 2003b, pp.110-113.

Bouzouba, K. and Moulin, B. Connaissances implicites et sociales : dialogisme des interactions discursives. In *Analyse et Simulation de Conversation : De la Théorie des Actes de Discours aux Systèmes Multi-Agents*, Moulin, B., Delisle, S., and Chaib-draa, B. (eds.), L'interdisciplinaire informatique, 1999, pp. 203-241.

Bouzouba, K., Bentahar, J., and Moulin, B. Dialogization and implicit information in an agent communicational model. In *Developments in Agent Communication*, van Eijk, R., Huget, M.P. and Dignum, F. (eds.), Int. Workshop on Agent Communication, AAMAS'04, LNAI 3396, Springer, 2004, pp. 211-226.

Bouzouba, K., Moulin, B. KQML+: An extension of KQML in order to deal with implicit information and social relationships. In Proc. of FLAIRS, 1998, pp. 289-293.

Bouzouba, K., Moulin, B., and Kabbaj, A. CG-KQML+: an agent communication language and its use in a multi-agent system. In Proc. of the 9th Int. Conf. on Conceptual Structures, 2001, pp. 1-14.

Bratman, M.E. *Intentions, plans and practical Reason*. Harvard University Press, Cambridge, 1987.

Brewka, G. Dynamic argument systems: A formal model of argumentation processes based on situation calculus. In *Journal of Logic and Computation*, vol. 11(2), 2001, pp. 257-282.

Broersen, J., Dastani, M., Huang, Z., Hulstijn, J., and van der Torre, L. The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In Proc. of the 5th Int. Conf. On Autonomous Agents (AA), ACM Press, 2001, pp. 9-16.

Bylander, E. Complexity results for planning. In Proc. of the 12th Int. Conf. on Artificial Intelligence, 1991, pp. 274-279.

Castelfranchi, C. Commitments: from individual intentions to groups and organizations. In Proc. of the 1st Int. Conf. On Multi-Agent Systems, 1995, pp. 41-48.

Chaib-draa, B and Dignum, F. Trends in agent communication language. In *Computational Intelligence*, vol. 18(2), 2002, pp. 89-101.

Chaib-draa, B. Industrial applications of distributed artificial intelligence. *Communications of the ACM*, vol. 38(11), 1995, pp. 47-53.

Chaib-draa, B., Labrie, M.A., Bergeron, M., and Pasquier, P. DIAGAL: an agent communication language based on dialogue games and sustained by social commitments. Submitted to the *Journal of Autonomous Agents and Multi-Agent Systems*, 2005.

Chellas, B.F. *Modal Logic: an Introduction*. Cambridge University Press, New York, 1980.

Chellas, B.F. Time and modality in the logic of agency. In *Studia Logica*, Kluwer, vol. 51, 1992, pp. 485-518.

Chopra, A., Singh, M.P. Nonmonotonic commitment machines. In *Advances in Agent Communication*, Dignum, F. (ed.), Int. Workshop on Agent Communication Languages, AAMAS'03, LNAI 2922, Springer, 2004, pp. 183-200.

Clark, H.H. and Haviland, S.E. Psychological processes in linguistic explanation. In *Explaining Linguistic Phenomena*, Cohen, D. (ed.)., 1974, 91-124.

Clark, H.H. *Using Language*. Cambridge University Press, 1996.

Clarke, E.M., Emerson, E.A. and Sistla, A.P. Automatic verification of finite-state concurrent systems using temporal logic specifications. In *ACM Transactions on Programming Languages and Systems*, vol. 8(2), 1986, pp. 244-263.

Clarke, E.M., Grumberg, O. and Peled, D.A. *Model Checking*. The MIT Press: Cambridge, MA, 2000.

Cleaveland, R. Tableau-based model checking in the propositional mu-calculus. In *Acta Informatica*, vol. 27(8), 1990, pp.725-747.

Cohen, P.R. and Levesque, H.J. Persistence, intentions and commitment. In *Intentions in Communication*, Cohen, P.R., Morgan, J., and Pollack, M.E. (eds.), MIT Press, Cambridge, 1990, pp. 33-69.

Cohen, P.R. and Perrault, C.R. Elements of a plan-based theory of speech acts. In *Cognitive Science*, vol. 3, 1979, pp. 177-212.

Colombetti, M. A commitment-based approach to agent speech acts and conversations. In Proc. of the Autonomous Agent Workshop on Conversational Policies, 4th Int. Conf. On Autonomous Agent, 2000, pp. 21-29.

Cost, R.S., Chen, Y., Finin, T., Labrou, Y. and Peng, Y. Using colored Petri nets for conversation modeling. In *Issues in Agent Communication*, Dignum, F., and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 178-192.

Courcoubetis, C., Vardi, M.Y., Wolper, P. and Yannakakis, M. Memory efficient algorithms for verification of temporal properties. In *Formal Methods in System Design*, vol. 1, 1992, pp. 275-288.

Cox, B., Tygar, J. and Sirbu, M. Netbill security and transaction protocol. In Proc. of the 1st USENIX Workshop on Electronic Commerce, 1995, pp. 77-88.

Dastani, M., Hulstijn, J. and der Torre, L. V. Negotiation protocols and dialogue games. In Proc. of the Belgium/Dutch AI Conf., 2000, pp. 13-20.

De Giacomo, G., Lespérance, Y. and Levesque, H. ConGolog, a concurent programming language based on the situation calculus. In *Artificial Intelligence*, vol. 121, 2000.

Dignum, F. and Greaves, M. Issues in agent communication: an introduction. In *Issues in Agent Communication*, Dignum, F. and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 1-16.

Dignum, F., Dunin-Keplicz, B. and Verbugge, R., Agent theory for team formation by dialogue. In Proc. of the 7th Int. Workshop on Agent Theories, Architectures, and Languages, Castelfranchi, C. and Lespérence Y. (eds.), LNAI 1986, Springer, 2000, pp. 150-166

Dignum, F., Dunin-Keplicz, B., and Verbugge, R., Creation collective intention through dialogue. In *Logic Journal of the IGPL*, vol. 9(2), 2001, pp. 305-319.

Dignum, V., Meyer, J.-J.Ch., Dignum, F. and Weigand, H. Formal specification of interaction in agent societies. Formal Approaches to Agent-based Systems. In Proc. of FAABS 2002, Hinchey, M.G., Rash, J.L., Truszkowski, W.F., Rouff, C., and Gordon-Spears, D. (eds.), Springer, 2003, pp. 37-52.

Dung, P.M. On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. In *Artificial Intelligence*, vol. 77, 1995, pp. 321-357.

Elvang-Goransson, M., Fox, J. and Krause, P. Dialectic reasoning with inconsistent information. In Proc. of the 9th Conf. on Uncertainty in Artificial Intelligence, 1993, pp. 114-121.

Emerson, A.E. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, vol. B, 1990, pp. 995-1072.

Emerson, E.A. and Halpern, J.Y. Sometimes and not never, revisited: on branching versus linear time temporal logic. In *Journal ACM*, vol. 33(1), 1986, pp. 151-178.

Emerson, E.A. and Lei C.-L. Efficient model checking in fragments of the propositional mu-calculus. In Proc. of the 1st Annual Sym. on Logic in Computer Science, 1986, pp. 267-278.

Emerson, E.A. and Sistla, A.P. Deciding full branching time logic. In *Information and Control*, vol. 61, 1984, pp. 175-201.

Emerson, E.A. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, van Leeuwen, J. (ed.), vol. B, 1990, pp. 995-1072.

Emerson, E.A., Jutla, C. and Sistla, A.P. On model-checking for fragments of μ-calculus. In *Computer Aided Verification*, Courcoubetis, C. (ed.), LNCS 697, 1993, pp. 385-396.

Endriss, U., Maudet, N., Sadri, F., and Toni, F. Protocol conformance for logic-based agents. In Proc. of the 18th Int. Joint Conf. on Artificial Intelligence, 2003, pp. 679-684.

Finin, T., Labrou, Y. and Mayfield, J. KQML as an agent communication language. In *Software Agent*, Bradshaw, J.M. (ed.), AAAI Press / The MIT Press, 1995, pp. 291-315.

FIPA (1997, 1999, 2001a, 2002). Fipa-acl specifications: Foundation for intelligent physical agents. http://www.fipa.org/repository/aclspecs.php3.

FIPA-ACL. Communicative act library specification. Technical Report XC00037H, . Foundation for Intelligent Physical Agents, 2001b.

FIPA Interaction Protocols (2001, 2002). http://www.fipa.org/repository/ips.php3.

Fitting, M. Bilattices and the semantics of logic programming. In *Journal of Logic Programming*, vol. 11, 1991, pp. 91-116.

Flores, R.A., Pasquier P., and Chaib-draa, B. Conversational semantics with social commitments. In *Developments in Agent Communication*, van Eijk, R., Huget, M.P. and Dignum, F. (eds.), Int. Workshop on Agent Communication, AAMAS'04, LNAI 3396, Springer, 2004, pp. 19-36. Also in the *Journal of Autonomous Agents and Multi-Agent Systems* (to appear).

Flores, R.F. and Kremer, R.C. A formal theory for agent conversations for actions. In *Computational Intelligence*, 2002, (in press).

Fornara, N. and Colombetti, M. Defining protocols using a commitment-based agent communication language. In Proc. Of the 2nd Int. J. Conf. On Autonomous Agent and Multi-Agent Systems (AAMAS 03), ACM Press, 2003, pp. 520-527.

Fornara, N. and Colombetti, M. Operational specification of a commitment-based agent communication language. In Proc. Of the 1st Int. Joint Conf. on Autonomous Agent and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp. 535-542.

Fornara, N. and Colombetti, M. Protocol specification using a commitment based ACL. In Dignum, F. (ed.). *Advances in Agent Communication*. Int. Workshop on Agent Communication Languages, LNAI 2922, Springer, 2004, pp. 108-127.

Fox, J., Krause, P., and Ambler, S. Arguments, contradictions, and practical reasoning. In Proc. of the 10th European Conf. On Artificial Intelligence, 1992, pp. 623-626.

Giordano, L., and Martelli, A., On-the-fly automata construction for dynamic linear time temporal logic. In Proc. of 11th Int. Sym. On Temporal Representation and Reasoning (Time'04), 2004, pp. 133-139.

Giordano, L., Martelli, A., and Schwind, C. Verifying communicating agents by model checking in a temporal action logic. In *Logics in Artificial Intelligence* (JELIA'04), LNAI 3229, Springer, 2004, pp. 57-69.

Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N., and Turner, H. Nonmonotonic Causal theories. In *Artificial Intelligence*, 2003 (in press).

Grasso, F. A mental model for a rhetorical arguer. In Proc. of the European Cognitive Science Society Conf. Schmalhofer, F., Young, R., and Katz, G. (eds.), LEA, 2003.

Grasso, F. Towards a framework for rhetorical argumentation. In Proc. of the 6th Workshop on the Semantics and Pragmatics of Dialogue (EDILOG'02), 2002, pp. 53-60.

Greaves, M., Holmback, H., and Bradshaw, J. What is a conversation policy? In *Issues in Agent Communication*. Dignum. F. and Greaves. M. (eds.), LNAI 1916, Springer, 2000, pp. 118-131.

Grice, H.P. Logic and Conversation. In *Speech Acts: Syntax and Semantics*, Cole, P. and Morgan, J. (eds.), vol. 11. New York: Academic Press, 1975, pp. 41-58.

Grice, H.P. Meaning. In *Basic Topics in the Philosophy of Language*. Harnish, R.M. (ed.), Harvester Wheatsheaf, 1957, pp. 21-29.

Grosz, B.G. and Sidner, C.L. Plans for discourse. In *Intentions in Communication*, Cohen, P.R., Morgan, J., and Pollack, M.E. (eds.), MIT Press, Cambridge, 1990, pp. 417-444.

Grosz, B.J. and Kraus, S. Collaborative plans for complex group action. In *Artificial Intelligence*, vol. 86(2), 1996, pp. 269-357.

Guerin, F. and Pitt, J. Denotational semantics for agent communication languages. In Proc. of the 5th Int. Conf. on Autonomous Agents, Müller, J.P., Andre, E., Sen, S. and Frasson, C. (eds.), ACM Press, 2001, pp. 497-504.

Günter, A. Some ways of representing dialogues. In *Cognitive Constraints on Communication*, Vatina, L. and Hintikka, J. (eds.), 1984, pp. 241-250.

Habermas, J. *The Theory of Communicative Action*, vol. 1 and 2, Polity Press, Cambredge, UK, 1984.

Hafer, T. and Thomas, W. Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree. In Proc. of 14th Int. Collogue on Automata, Languages and Programming, LNCS 267, Springer, 1987, pp. 269-279.

Hamblin, C.L. *Fallacies*. Methuen, 1970.

Hamblin, C.L. Mathematical models of dialogue. In *Theoria*, vol.37, 1971, pp. 130-155.

Harel, D. Dynamic logic. In *Handbook of Philosophical Logic*, Gabbay, D.M. and Guenther, F. (eds.), vol. 2, 1984, pp. 497-604.

Harel, D. Dynamic logic: axiomatics and expressive power. Ph.D. Thesis, MIT, LNCS 68, 1979.

Henriksen, J.G. and Thiagarajan. Dynamic linear temporal time logic. In *Annals of Pure and Applied Logic*, vol. 96, 1999, pp. 187-207.

Herrestad, H. and Krogh, C. Obligations directed from bearers to counterparties. In Proc. of 5th Int. Conf. on Artificial Intelligence and Law, 1995, pp. 210-218.

Hindriks, K.V., de Boer, F.S., van der Hoek, W., and Meyer, J.-J.Ch. Semantics of communication agents based on deduction and abduction. In *Issues in Agent Communication*, Dignum, F. and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 63-79.

Hintikka, J. *Knowledge and Belief*. Cornell Univesity Press, 1962.

Hintikka, J. The modes of modality. *Acta Philosophica Fennica*, vol.16, 1963, pp. 65-81.

Holzman, G.J. The model checker Spin. *IEEE Transaction on Software Engineering*, vol. 23(5), 1997, pp. 279-295.

Huber, M.J., Kumar, S., and McGee, D. Toward a suite of performatives based upon joint intention theory. In In *Developments in Agent Communication*, van Eijk, R., Huget, M.P. and Dignum, F. (eds.), Int. Workshop on Agent Communication, AAMAS'04, LNAI 3396, Springer, 2004, pp. 245-260.

Huber, M.J., Kumar, S., Cohen, P.R., and McGee, D. A formal semantics for proxy communicative acts. In Proc. of Agent Theories, Architectures, and Languages (ATAL 01), 2001, pp. 221-234.

Huget, M.-P., Wooldridge, M. Model checking for ACL compliance verification. In *Advances in Agent Communication*, Dignum, F. (ed.), Int. Workshop on Agent Communication Languages, LNAI 2922, Springer, 2004, pp. 75-90.

Huhns, M.N and Singh, M.P. *Readings in Agents*. Morgan Kaufmann, San Francisco, California, 1998.

Hulstijn, J. Dialogue games are recipe for joint action. In Proc. of the 4th Workshop on Formal Semantics and Pragmatics of Dialogue, 2000b.

Hulstijn, J. Dialogue Models for Inquiry and Transaction. PhD Thesis, University Twente, Enschede, The Netherlands, 2000a.

Huth, M.R.A. and Ryan, M.D. *Logic in Computer Science. Modeling and Reasoning about Systems*. Cambridge University Press, 2000.

Kacprzak, M. and Penczek, W. Unbounded model checking for alternating-time temporal logic. In Proc. of the 3rd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems, 2004a, pp. 646-653.

Kacprzak, M., Lomuscio, A., and Penczek, W. Verification of multiagent systems via unbounded model checking. In Proc. of the 3rd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems, 2004b, pp. 638-645.

Khan, S. and Lespérance, Y., A model of rational agency for communicating agents. In *Developments in Agent Communication*, van Eijk, R., Huget, M.P. and Dignum, F. (eds.), Int. Workshop on Agent Communication, AAMAS'04, LNAI 3396, Springer, 2004, pp. 261-280.

Kone, M.T., Shimazu, A., and Nakajima, T. The state of the art in agent communication languages. In *Knowledge and Information Systems*, 2000, pp. 259-284.

Kowalski, R. and Sergot, A. A logical-based calculus of events. In *New Generation Computing*, vol. 4(1), 1986, pp. 67-95.

Krause, P., Ambler, S., Elvang-Gorannson, M., and Fox, J. A logic for argumentation for reasoning under uncertainty. In *Computational Intelligence*, vol. 11(1), 1995, pp. 113-131.

Kripke, S. Semantical analysis of modal logic. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, vol. 9, 1963, pp. 67-96.

Kumar, S., Huber, M.J., McGee, D.R., Cohen, P.R., Levesque, H.J. Semantics of agent communication languages for group interaction. In Proc. of the 16th Nat. Conf. on Artificial Intelligence, pp. 42-47, 2000.

Labrie, M.A., Chaib-draa, B., and Maudet, N. DIAGAL: a tool for analyzing and modelling commitment-based dialogues between agents. In Proc. of 16th Canadian Conference on Artificial Intelligence, Xiang, Y. and Chaib-draa, B. (eds.), LNAI 2671, Springer, 2003, pp. 353-369.

Labrou, Y. and Finin, T. Semantics and conversation for an agent communication language. In *Readings in Agents*, Huhns, M. and Singh, M.P (eds.), Morgan Kaufman, 1998, pp. 235-242.

Labrou, Y. Semantics for an agent communication language. Ph.D. Thesis, University of Maryland, USA, 1997.

Labrou, Y., Finin, T., and Peng, Y. Agent communication languages: the current landscape. In *IEEE Intelligent Systems*, 1999, pp.45-52.

Lebbink, H-J., Witteman, C.L.M., and Meyer, J-J. Ch. Dialogue games for inconsistent and biased information. In *Electronic Notes in Theoretical Computer Science*, vol. 85(2), 2004.

Lespérence, Y. On the epistemic feasibility of plans in multi-agent systems specifications. In *Intelligent Agents VIII*, Meyer, J-J.Ch. and Tambe, M. (eds.). Proc. of the Workshop on Agent Theories, Architectures, and Languages (ATAL-2001), LNAI 2333, Springer, 2002, pp. 69-85.

Lichtenstein, O. and Pnueli, A. Checking that finite state concurrent programs satisfy their linear specification. In 12th Annual ACM Sym. On Principles of Programming Languages, 1985, pp. 97-107.

Litman, D.J and Allen, J.F. Discourse processing and commonsense plans. In *Intentions in Communication*, Cohen, P.R, Morgan, J., and Pollack, M.E. (eds.), MIT Press, Cambridge, 1990, pp. 365-388.

Lorenzen, P. Logik und agon. In Atti del XII Congresso Internazionale di Folosophia. IV: Logica, Linguaggio e Communicazione, 1960, pp. 187-194.

MacKenzie, J.D. Four dialogue systems. In *Studia*, vol.49(4), 1990, 567-583.

MacKenzie, J.D. Question-begging in non-cumulative systems. In *Journal Of Philosophical Logic*, vol. 8, 1979, pp. 117-133.

Mallya, A.U., Yolum, P. and Singh, M.P. Resolving Commitments Among Autonomous Agents. In *Advances in Agent Communication*, Dignum, F. (ed.), Int. Workshop on Agent Communication Languages, LNAI 2922, Springer, 2004, pp. 166-182.

Maudet, N. and Chaib-draa, B. Commitment-based and dialogue-game based protocols, new trends in agent communication languages. In *Knowledge Engineering Review*, vol. 17(2), Cambridge University Press, 2002, pp. 157-179.

Maudet, N. Modéliser les conventions des interactions langagière : la contribution des jeux de dialogue. PhD Thesis of Université Paul Sabatier-Toulouse, France, 2001.

Maudet, N., Chaib-draa, B., and Labrie, M.A. Request for action reconsidered as dialogue game based on commitments. In *Communication in Multiagent Systems*, Huget, M.-P. (ed.), Int. Workshop on Agent Communication Languages and Conversation Policies, AAMAS 02, LNAI 2650, Springer, 2002, pp. 284-299.

McBurney, P. and Parsons, S. Agent ludens: games for agent dialogues. In Proc. of the AAAI Spring Sym. on Game Theoretic and Decision Theoretic Agents, 2001, pp. 70-77.

McBurney, P. and Parsons, S. Games that agents play: A formal framework for dialogues between autonomous agents. In *Journal of Logic, Language and Information*, vol. 11(3), 2002, pp. 315-334.

McBurney, P. Rational interaction. PhD. Thesis of University of Liverpool, UK, 2002.

McBurney, P., Parsons, S. Risk agoras: Dialectical argumentation for scientific reasoning. In Proc. of the 16th Conf. On Uncertainty in Artificial Intelligence, Morgan Kaufmann, 2000, pp. 371-379.

McBurney, P., Parsons, S., and Wooldridge, M. Desiderata for agent argumentation protocols. In Proc. of the 1st Int. Joint Conf. On Autonomous Agent and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp 402-409.

McCartyh, J. and Hayes, P. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, vol. 4, 1969, pp. 463-502.

McMillan, K.L. Aplying SAT methods in unbounded symbolic model checking.. In *Computer Aided Verification*, Brinksma, E., Guldstrand Larsen, K. (eds.), LNCS 2404, Springer, 2002, pp. 250-264.

Meyer, J-J. Ch., van der Hoek, W. and van Linder, B. A logical Approach to the dynamics of commitments. In *Artificial Intelligence Journal*, vol. 113 (1-2), 1999, pp. 1-40.

Moore, R.C. Reasoning about knowledge and action. Technical report 191, SRI International, 1980.

Morgenstern, L. A first order theory of planning, knowledge, and action. In Proc. of the the 1st Conf. On Theoretical Aspects of Reasoning about Knowledge (TARK 86), Halpern, J.Y. (ed.), Morgan Kaufmann, 1986, pp. 99-114.

Morgenstern, L. Knowledge preconditions of actions and plans. In Proc. of the 10th Int. Joint Conf. On Artificial Intelligence (IJCAI 87), 1987, pp. 867-874.

Morris, C.W. Foundations of the theory of signs. In *Int. Encyclopedia of Unified Science*, Neurath, O., Carnap, R., and Morris, C.W. (eds.).Chicago University Press, 1938, pp. 77-138.

Moulin, B. The social dimension of interactions in multi-agent systems. In Wobcke, W., Pagnucco, M. and Zhang, W. (eds.). *Agent and Multi-Agent Systems, Formalisms, Methodologies and Applications*, Artificial Intelligence 1441, 1998, pp. 109-122.

Moulin, B., and Chaib-draa, B. Distributed artificial intelligence: an overview. In *Foundations of Distributed Artificial Intelligence*, Jennings, N. and O'Hare, G. (eds.), Wiley, 1996, pp. 3-55.

Muller, D.E. and Schupp, P.E. Alternating automaton on infinite trees. In *Theoretical Computer Science*, vol. 54, 1987, pp. 267-276.

Muller, D.E., Saoudi, A. and Schupp, P.E. Weak alternating automata gives a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In Proc. of the 3rd IEEE Sym. on Logic in Computer Science, 1988, pp. 422-427.

Parsons, S. Normative argumentation and qualitative probability. In. Proc. Of the 1st Int. Joint Conf. On Qualitative and Quantitative Practical Reasoning. Gabay, D.M., Kruse, R., Nonnengart, A., and Ohlbach, H.J. (eds.), LNAI 1244, Springer, 1997.

Parsons, S., Jennings, N.R. Negotiation through argumentation-a preliminary report. In Proc. of the 2nd Int. Conf. On Multi Agent Systems, 1996, pp. 267-274.

Parsons, S., Wooldridge, M. and Amgoud, L. An analysis of formal inter-agent dialogues. In Proc. of the 1st Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp. 394-401.

Parsons, S., Wooldridge, M. and Amgoud, L. On the outcomes of formal inter-agent dialogues. In Proc. of the 2nd Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 03), ACM Press, 2003, pp. 616-623.

Pasquier, P. and Chaib-draa, B. The cognitive coherence approach for agent communication pragmatic. In Proc. of the 2nd Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 03), ACM Press, 2003, pp. 544-551.

Pasquier, P., Andrillon, N., Chaib-draa, B., and Labrie, M.A. An exploration in using cognitive coherence theory to automate BDI agents' communicational behavior. In *Advances in Agent Communication*, Dignum, F. (ed.), Int. Workshop on Agent Communication Languages, LNAI 2922, Springer, 2003, pp. 37-58.

Penczek, W. and Lomuscio, A. Verifying epistemic properties of multi-agent systems via model checking. In *Fundamenta Informaticae*, vol. 55(2), 2003, pp. 167-185.

Perelman, C. and Olbrechts-Tyteca, L. La *Nouvelle Rhétorique : Traité de l'Argumentation*. Presses Universitaire de France, 1958. Translated on *The New Rhetoric: a Treatise on Argumentation*. University of Notre Dame Press, Indiana, 1969.

Perrault, C.R. and Allen, J.F. A plan-based analysis of indirect speech acts. In *American Journal of Computational Linguistics*, vol. 6(3-4), 1980, pp. 167-182.

Pitt, J. and Mamdani, A. Communication protocols In multi-agent systems: a development method and reference architecture. In *Issues in Agent Communication*, Dignum, F. and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 160-177.

Pnueli, A. Application of temporal logic to the specification and verification of reactive systems: a survey of current trends. In *Current Trends in Concurrency: Overviews and Tutorials*, de Bakker, J.W. de Roever, W.P. and Rozenberg, G. (eds.), LNCS 224, Springer, 1986.

Pollack, M.E. Plans as complex mental attitudes. In *Intentions in Communication*. Cohen, P.R., Morgan, J., and Pollack, M.E. (eds.), MIT Press, Cambridge, 1990, pp. 77-104.

Pollock, J.L. A theory of defeasible reasoning. In *Int. Journal of Intelligent Systems*, vol.6, 1991, pp. 33-54.

Pollock, J.L. How to reason defeasibly. In *Artificial Intelligence*, vol. 57, 1992, pp 1-42.

Prakken, H. and Sartor, G. A dialectical model of assessing conflicting arguments in legal reasoning. In *Artificial Intelligence and Law*, vol. 4, 1996, pp 331-368.

Prakken, H. and Sartor, G. Modelling reasoning with precedents in a formal dialogue game. In *Artificial Intelligence and Law*, vol.6, 1998, pp. 231-287.

Prakken, H. and Vreeswijk, G. Logics for defeasible argumentation. In *Handbook of Philosophical Logic*, Gabbay, D. (ed.), Second Edition, Kluwer, 2000.

Prakken, H. Logical Tools for Modelling legal argument. A study of defeasible reasoning in law, *Kluwer Law and Philosophy Library*, 1997.

Prakken, H. Relating protocols for dynamic dispute with logics for defeasible argumentation. In *Synthese* (127), 2001, pp. 187-219.

Prakken, H., Reed, C.A., and Walton, D.N. Argumentation schemes and generalisations in reasoning about evidence. In Proc. of the 9th Int. Conf. On Artificial Intelligence and Law, ACM Press, 2003, pp. 32-41.

Pulman, S.G. Conversational games, belief revision and bayesian networks. In Proc. of 7th Computational Linguistics in the Netherlands Meeting, 1996, pp. 1-25.

Queille, J.P. and Sifakis, J. Specification and verification of concurrent systems in Cesar. In Proc. of 5th Int. Sysm. on Programming, LNCS 137, Springer, 1981, pp. 337-351.

Rahwan, I., Ramchurn, S.D., Jennings, N.R., McBurney, P., Parsons, S., and Sonenberg, L. Argumentation-based negotiation. In *Knowledge Engineering Review*, 2004, pp. 343-375.

Raimondi, F. and Lomuscio, A. Verification of multiagent systems via ordered binary decision diagrams: an algorithm and its implementation. In Proc. of the 3rd Int. Joint Conf.

on Autonomous Agents and Multi Agent Systems (AAMAS 04), ACM Press, 2004, pp. 630-637.

Ramchurn, S.D., Sierra, C., Jennings, N.R., and Godo L. A computational trust model for multi-agent interactions based on confidence and reputation. In Proc. of 6th Int. Workshop of Deception, Fraud and Trust in Agent Societies, 2003, pp. 69-75.

Rao, A.S. AgentSpeak(L): BDI agents speak out in a logical computable language. In Proc. of the 7th European Workshop on Modeling Autonomous Agents in a Multi-Agent World, Van de Velde, W. and Perram, J.W. (eds.), LNAI 1038, Springer, 1996, pp. 42-55.

Rao, A.S. and Georgeff, M.P. A model-theoretic approach to the verification of situated reasoning systems. In Proc. of the 13th Int. Joint Conf. on Artificial Intelligence (IJCAI-93), 1993, pp. 318-324.

Rao, A.S. and Georgeff, M.P. BDI agents: from theory to practice. In Proc. of the 1st Int. Conf. on Multi-Agent Systems (ICMAS 95), 1995, pp. 312-319.

Rao, A.S. and Georgeff, M.P. Modeling rational agents within a BDI-architecture. In Proc. of the 2nd Int. Conf. On Principles of Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann, Cambridge, 1991, pp. 473-484.

Reed, C. Dialogue frames in agent communication. In Proc. of the 3rd Int. Conf. on Multi-Agent Systems, 1998, pp. 246-253.

Reed, C.A. and Rowe, G.W.A. Araucaria: software for puzzles in argument diagramming and XML. Department of Applied Computing, University of Dundee. Technical Report, 2001.

Reed, C.A. and Walton, D.N. Argumentation schemes in argument-as-process and argument-as-product. In Proc. of the Conf. Celebrating Informal Logic @25, Canada, 2003.

Rousseau, D., Moulin, B., and Lapalme, G. Interpreting communicative acts and building a conversation model. In *Journal of Natural Language Engineering*, Cambridge, vol. 2(3), 1996, pp. 253-276.

Sabater, J. and Sierra, C. Reputation and social network analysis in multi-agent systems. In Proc. Of the 1st Int. J. Conf. On Autonomous Agents and MultiAgent Systems (AAMAS 02), ACM Press, 2002. pp. 475-482.

Sadek, M.D. Attitudes mentales et interaction rationnelle : vers une théorie formelle de la communication. Thesis of Université de Rennes I, France, 1991.

Sadek, M.D., Bretier, P., and Panaget, F. ARTIMIS: natural dialogue meets rational agency. In Proc. of the 14th Int. J. Conf. on Artificial Intelligence (IJCAI-97), Morgan, P., Publisher, 1997, pp. 1030-1035.

Sadri, F., Toni, F., and Torroni, P., Logic agents, dialogues and negotiation: an abductive approach. In Proc. of the Sym. on Information agents for E-Commerce., Artificial Intelligence and the Simulation of Behaviour Conf., 2001.

Searle, J.R. and Vanderveken, D. *The Foundations of Illocutionary Logic*. Cambridge University Press, 1985.

Searle, J.R. *Intentionality: an Essay in the Philosophy of Mind*. Cambridge University Press, Cambridge, England, 1983.

Searle, J.R. *Speech Acts: an Essay in the Philosophy of Language*. Cambridge University Press: Cambridge, England, 1969.

Shanahan, M. An abductive event calculus planner. *In Journal of Logic Programming*, vol. 44, 2000, pp. 207-239.

Shapiro, S. and Lespérance, Y. Modeling multi-agent systems with the cognitive agents specification language - A feature interaction resolution application. In *Intelligent Agents Volume VII*, Castelfranchi, C. and Lespérance, Y. (eds.), Proc. of the Workshop on Agent Theories, Architectures, and Languages (ATAL-2000), LNAI 1986, Springer, 2001, pp. 244-259.

Shapiro, S., Lespérance, Y. and Levesque, H.J. Specifying communicative multi-agent Systems. In *Agents and Multi-Agent Systems - Formalisms, Methodologies, and Applications*, Wobcke, W., Pagnucco, M. and Zhang, C. (eds.), LNAI 1441, Springer, 1998, pp. 1-14.

Shapiro, S., Lespérance, Y. and Levesque, H.J. The cognitive agents specification language and verification environment for multi-agent systems. In Proc. of the 1st Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS-02), ACM Press, 2002, pp. 19-26.

Sierra, C. Jennings, N.R., Noriega, P., and Parsons, S. A framework for argumentation-based negotiation. In *Intelligent Agents IV*, Singh, M.P, Rao, A., Wooldridge, M. (eds.), LNAI 1365, Springer, 1998, pp. 177-192.

Simari, G.R. and Loui, R.P. A mathematical treatment of defeasible reasoning and its implementation, In *Artificial Intelligence*, vol. 53, 1992, pp 125-157.

Singh, M.P. A social semantics for agent communication language. In *Issues in Agent Communication*, Dignum, F. and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 31-45.

Singh, M.P. Agent communication languages: rethinking the principles. In *IEEE Computer*, 1998, pp. 40-47.

Singh, M.P. An ontology for commitments in multi-agent systems: toward a unification of normative concepts. In *Artificial Intelligence and Law*, vol. 7, 1999, pp. 97-113.

Singh, M.P., Rao, A.S. and Georgeff, M.P. Formal methods in DAI: logic-based representation and reasoning. In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, Weiss, G. (ed.), MIT Press, 1999, pp. 331-376.

Smith, R.G. The contract net protocol: High-level communication and control in a distributed problem solver. In *IEEE Transactions on Computers*, vol. 29, 1980, pp. 1104-1113.

Sowa, J.F. Conceptual Structures: *Information Processing in Mind and Machines*. Addison-Wesley, Reading, MA, 1984.

Spade, P.V., Recent research on medieval logic. In *Synthese*, vol.40, 1979, pp. 3-18.

Stirling, C. and Walker, D. Local model checking in the modal mu-calculus. In *Theoretical Computer Science*, vol. 89(1), 1991, pp. 161-177.

Sycara, K. Multiagent Systems. In *AI Magazine*, American Association for Artificial Intelligence, vol. 19(2), 1998, pp. 79-92.

Sycara, K. Persuasion argumentation in negotiation. In *Theory and Decision*, vol. 28, 1990, pp. 203-242.

The Agent Oriented Software Group. Jack 4.1, 2004. www.agent-software.com/

Tohmé, F. Negotiation and defeasible reasons for choice. In Proc. of the Stanford Spring Sym. On Qualitative Preferences in Deliberation and Practical Reasoning, 1997, pp. 95-102.

Toulmin, S. *The Uses of Argument*. Cambridge University Press, Cambridge, England, 1958.

Traum, D. A reactive-deliberative model of dialogue agency. In Proc. of Intelligent agents iii, Agent Theories, Architectures, and Languages, 1996, pp. 157-172.

van der Hoek, W. and Wooldridge, M. Model checking knowledge and time. In *Model Checking Software*, LNCS 2318, Springer, 2002, pp. 95-111.

van der Hoek, W. and Wooldridge, M. Towards a logic of rational agency. In *Logic Journal of the IGPL*, vol. 11(2), 2003, pp. 133-157.

van Eemeren, F.H., Grootendorst, R., Henkemans, F.S., Blair, J.A., Johnson, R.H., Krabbe, E.C.W., Plantin, C., Watton, D.N., Willard, C.A., Woods, J., and Zarefsky, D. *Fundamental of Argumentation Theory*. Lawrence Erlbaum Associates, Mahwah, USA, 1996.

van Eijk, R.M., de Boer, F.S., van der Hoek, W., and Meyer, J.-J.Ch. A verification framework for agent communication. In the *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 6, 2003, pp. 185-219.

van Eijk, R.M., der Boer, F.S., van der Hoek, W., and Meyer, J.-J.Ch. On dynamically generated ontology translators in agent communication. In *International Journal of Intelligent Systems*, vol.16(5), 2001, pp.587 .607.

van Eijk, R.M., de Boer, F.S., van der Hoek, W., and Meyer, J.-J.Ch. Operational semantics for agent communication languages. In *Issues in Agent Communication*, Dignum, F. and Greaves, M. (eds.), LNAI 1916, Springer, 2000, pp. 80-95.

van Linder, B. van der Hoek, W. and Meyer, J.-J.Ch. Formalizing abilities and opportunities of agents. In *Fundamenta Informaticae*, vol. 34(1, 2), 1998, pp. 53-101.

Vanderveken, D. Illocutionary logic and discourse typology. In Special Issue 216 Searle with his Replies of *Revue Internationale de Philosophie*, Vanderveken, D. (ed.), 2001, pp. 243-255.

Vardi, M. and Wolper, P. An automata-theoretic approach to automatic program verification. In Sym. on Logic in Computer Science, 1986, pp. 332-344.

Verdicchio, M. and Colombetti, M. A logical model of social commitment for agent communication. In Proc. of The Second Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 03), ACM Press, 2003, pp. 528-535.

Verdicchio, M. and Colombetti, M. Commitment for agent-based supply chain management. In *ACM SIGecom Exchanges*, vol. 3(1), 2002, pp. 13-23.

Visser, W., Havelund, K., Brat, G., and Park, S. model checking programs. In Proc. of the 5th Int. Conf. on Automated Software Engineering, IEEE Computer Society, 2000, pp. 3-12.

Vongkasem, L. and Chaib-draa, B. ACL as a joint project between participants: a preliminary report. In *Issues in Agent Communication*, Dignum. F., and Greaves. M. (eds.), LNAI 1916, Springer, 2000, pp. 31-45.

Vreeswijk, G.A.W. Abstract argumentation systems. In *Artificial Intelligence*, vol. 90, 1997, pp. 225-279.

Walton, D.C. Model Checking Agent Dialogues. In Proc. of 2nd Int. Workshop on Declarative Agent Languages and Technologies, 2004.

Walton, D.N and Reed, C.A. Diagramming, argumentation schemes, and critical questions. In *Anyone Who Has a View: Theoretical Contributions to the Study of Argumentation*. Van Eemeren, F.H., Blair, J.A., Willard, C.A., and Snoek Henkemans, A.F. (eds.), Kluwer, Dordrecht, 2003, pp. 195-211.

Walton, D.N. and Krabbe, E.C.W. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, NY, 1995.

Winograd, T. and Flores, F. Understanding computers and cognition: A new foundation for design. Albex Publishing Co., Norwood, USA, 1986.

Wittgenstein, L. *Philosophical Investigations*. Oxford: Basil Blackwell, 1958.

Wolper, P. On the relation of programs and computations to models of temporal logic. In Proc. of the Temporal Logic in Specification, Banieqbal, B., Barringer, H. and Pneuli, A. (eds.), LNCS 1989, Springer, pp. 75-123.

Wooldridge, M. and Jennings, N.R. Intelligent agents: theory and practice. In *The Knowledge Engineering Review*, vol. 10(2), 1995, pp. 115-152.

Wooldridge, M. Fisher, M., Huget, M.P. and Parsons, S. Model checking multi-agent systems with MABLE. In Proc. of the 1st Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp. 952-959.

Wooldridge, M. *Reasoning about Rational Agents*. The MIT Press: Cambridge, MA, 2000.

Wooldridge, M. Semantic issues in the verification of agent communication languages. In *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 3(1), Kluwer, 2000, pp. 9-31.

Yolum, P., Singh, M.P. Flexible protocol specification and execution: applying event calculus planning using commitments. In Proc. of the 1st Int. J. Conf on Autonomous Agents and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp. 527-534.

Yu, B. and Singh, M. An evidential model of distributed reputation management. In Proc. Of the 1st Int. J. Conf. On Autonomous Agents and Multi-Agent Systems (AAMAS 02), ACM Press, 2002, pp. 294-301.