UNIVERSITY OF LIVERPOOL

Argumentation-based Dialogues over Cooperative Plans

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of
Doctor of Philosophy
by

Angel Rolando Medellin Gasque

August 2013

A mi familia en general y a mis padres en particular.

Contents

Illustrations

List of Figures

List of Tables

Abstract

Argumentation-based Dialogues over Cooperative Plans
by Angel Rolando Medellin Gasque

If autonomous agents operating with other agents in open systems are to fulfil their goals and design objectives, the need to discuss and agree upon plans of action is imperative. In this thesis I present work covering both theoretical research and practical development related to the use of *argumentation-based dialogues* as a way to coordinate actions in multi-agent planning scenarios.

The necessity of *coordination* in multi-agent systems requires the development of mechanisms to propose, modify, share, monitor, and argue about plans. In this thesis I present an argumentation scheme to propose multi-agent plans and associated critical questions to critique the proposal. Such a detailed consideration of multi-agent plan composition contains the right characteristics to enable the justification of plans.This research builds upon research on practical reasoning for action proposals and considers multi-agent plan proposals where plans require several agents for their execution.

A dialogue game protocol is also presented which is based on proposal framework. The protocol allows agents to engage in dialogues to *agree on* and modify plans based on persuasion and deliberation protocols. The detail encompassed by the argumentation scheme and critical questions means that there is a large number of critical questions, and so dialogues may be very lengthy. To overcome this issue, I investigated the issue of strategies for use with this dialogue game in terms of the different possible orderings in which critiques can be posed. The thesis presents an implementation that realises the theoretical framework in terms of a agents engaging in simulated dialogues to share and agree on a plan. The experiments allow us to investigate the effects of such strategies in terms of the number of questions issued to reach an agreement.

Overall, the framework presented in this thesis allow agents to engage in dialogues over cooperative plan proposals in a structured way using well-founded argumentative principles.

Acknowledgements

Doing a Ph.D. is satisfactory and at times a lonely process. But there is always someone to interact and share ideas. It is not until now, that I realise how many people have directly or indirectly influenced in my research. I would like to thank them all here in a symbolic way.

Firstly, my PhD was funded by the Mexican council of Science and Technology CONACYT to whom I am deeply grateful.

I had the fortune to have not two but three supervisors. The support, patience and guidance of Prof. Peter McBurney, Dr. Katie Atkinson and Prof. Trevor Bench-Capon was always precise and motivating. I cannot thank you enough for all your time and advice. It has been a pleasure for me to work with you.

When I joined the Department of Computer Science at the University of Liverpool as an MSc student back in 2007 I was told it was one of the most important and influential CS departments in England. I have no doubt about it. I would like to thank in no particular order, people who helped me in almost five years in Liverpool. Thanks to: Prof. Frans Coenen, Dr. Ullrich Hustadat, Dr. Clare Dixon, Dr. Valentina Tamma, Dr. Floriana Grasso, Prof. Michael Wooldridge, Prof. Lesek Gasienec, Dr. Adam Wyner, Prof. Paul Dunne, Mr. Dave Shield, Mr. Ken Chan, Mr. Phil Jimmieson, Ms Helen Bradley, Dr. Alexei Lisitsa, Prof. Paul Goldberg, Dr. Irina Biktasheva and all the staff in general.

My colleagues at the office were always supportive and friendly all the way long: Dr Omar Baqueiro Espinoza, Dr Elissabetta Erriquez, Dr Dan Cartwright, Dr Ji Ruan, Dr. Maya Wardeh, Dr. Andy Dowell, Dr. Nico Troquard, Ali Bojapur, Matias Garcia, Stephanie Chua, Luke Riley, Anton Minion, Martyn Lloyd Kelly and many more I am sure I am forgetting.

The Argumentation Community which I had the opportunity to meet in conferences always was encouraging, thanks to Dr. Sanjay Modgil, Prof. Chris Reed, Prof. Guillermo Garcia Simari, Dr. Francesca Toni, Prof. Henry Prakken, Prof. Simon Parsons, Dr. Vanina Garcia, Dr. Federico Cerutti, Dr. Nir Oren, Dr. Alice Toniolo, Dr. Xiyu Fan, Dr. Stella Heras, Prof. Michael Luck, Mr. Dionysios Kontarinis, Dr. Jodi Schenider.

I would like to thank my examiners Dr. Elizabeth Black and Prof. Wiebe van der Hoek. Their feedback and opinions on my work made me work even harder and I am nothing but grateful.

Support from friends in England was very important to me, thanks for the ride to Sergio, Nena, Lalo, Daniel, Elihu, Carlos, Ana, Fabian, Nele, Mariana, Elda y Carlos, Mauricio, Samuel, Augusto, Alicja, Juansethi, Claudia, Miguel.

Thanks to my beautiful friends Lindsey and Antoinette. Without you, these years would have been much less fun. Thanks for all the beer, the Irish music and the afternoons cheering to the mighty Liverpool FC.

To my beautiful girlfriend, soon to be wife, Elisa, you have been amazingly loving and supportive. Te amo (tanto!).

Finally, my family has always been there for me since I took the decision to live in England. I missed you all. The Bravo family, my warrior cousins. My beautiful sisters Lizbeth and Marisol. I'm sorry I have been away all these years. I am proud of you. My nephews Miguel Angel, Daniel, I'm sorry I missed your childhood, my beautiful niece Pamela I missed you every single day. My grandparents Rolando, Lizbeth, Mary thanks for all the support and advice. To my father for all his hard work and to my beautiful mother. You taught me invaluable things in life and I'm proud to be your son.

Chapter 1

Introduction

1.1 Motivation and Objectives

The demands of agent-based computing in uncertain and dynamic environments require planning agents to communicate and manage their plans and resources effectively [140]. The complexity of distributed systems limits the use of single-agent planning mechanisms in distributed scenarios because the local knowledge or capabilities of an agent are often not sufficient to generate a satisfactory plan. A common assumption in classical Artificial Intelligence (AI) planning is that the planner has accurate and complete knowledge of the world and the capabilities of other agents [63]. Since this assumption is rarely satisfied when using multi-agent systems, this thesis proposes the use of structured argumentation-based dialogues to coordinate the *proposal, selection and refinement* of multi-agent plans.

This thesis contributes to an area of AI research where the aim is to provide autonomous agents with mechanisms to communicate and cooperate when selecting and executing a plan in a non-deterministic environment [16, 128, 170, 174]. I present work related to the use of a structured *argumentation-based dialogue* based on an argumentation scheme for plan proposals and associated critical questions as a mechanism to support automated coordination in distributed planning scenarios. I will discuss the characteristics of such dialogues and examine some issues that present themselves when adapting existing work on practical reasoning for autonomous agents and argumentation dialogues. In the approach suggested, agents coordinate their beliefs and intentions using a dialogue game based on an argumentation scheme and a related set of critical questions.

The complexity[1] of dialogues to agree and modify plans makes particularly appropriate the use of argumentation schemes and a dedicated dialogue protocol. Argumentation schemes are suitable since they allow agents to engage in dialogues with the use predefined critical questions and present plans as a patterns of defeasible reasoning where the

[1]I use the term complexity in the sense of large, non-trivial and intricate relation of elements in the dialogue not in the sense of computational complexity.

arguments in favour of the plan can be arranged as the elements of the plan. Additionally the use of a dialogue game protocol allow agents to coordinate their intentions and apply their preferences exchanging only information relevant to the plans.

1.2 Research Questions

The research questions considered by this thesis are the following:

- What planning elements or features are relevant when agents discuss multi agent temporal plans?
- What argumentation scheme is appropriate for agents in order to allow them to engage in a dialogue about multi-agent plans?
- Which critical questions match the argumentation scheme for plan proposals to argue about plans?
- What type of protocol would be adequate to allow agents to engage in a dialogue about the critique and modification of multi-agent plans?
- How can agents identify, prioritize, and choose a relevant critical question in a dialogue about plans?
- How effective are the different strategies for choosing critical questions?

1.3 Overview of the Topic

1.3.1 The Problem

Multi-agent planning concerns with the problem of constructing a plan where several agents need to participate to reach a goal. This particular problem includes situations where a machine aids a human in constructing a plan (automated planning assistants) or multiple autonomous agents coordinate their activities towards a common goal [67]. There is a large variety of approaches to multi-agent planning [49] including distributed versions of classical AI planning techniques like NOAH [46], partial planning techniques [62], techniques that exploit specific multi-agent systems attributes like joint intentions [100], shared plans [81], plan merging [66], model checking approaches [43, 179], auction based approaches [205] and the one that is explored in this thesis, the argumentation-based dialogue approach [132].

Most work in Distributed AI has dealt with groups of agents pursuing common goals [37, 64] where interactions are guided by cooperation strategies meant to improve their common performance [167]. For agents, these agent interactions involve communicating plans and goals at an appropriate level of abstraction. A fair assumption in a cooperative scenario is that agents use the information shared to adjust their own local planning processes appropriately, so that the common planning goals are met, but conflicts can arise in complex domains so the intentions or actions of other agents need to be changed.

With self-interested agents, cooperative behaviour cannot be taken for granted. Cooperation has to emerge as a result of the agent interactions.

In multi-agent scenarios, agents can engage in negotiation dialogues to reach their objectives, although it is not always the case; with auctions for example, goals can be reached without negotiation or dialogue. In order to argue or negotiate effectively, agents need to (a) represent and maintain belief models, (b) reason about other agents' beliefs and goals, and (c) influence other agents' beliefs and behaviour [167]. Since an agent cannot be sure of the extent to which the other agent is willing to cooperate, the planning information communicated cannot simply be its own high level plans. It needs to take into consideration explicitly the beliefs and intentions of the recipient agent.

1.3.2 Proposed solution

If agents communicate their beliefs, intentions and preferences in a structured way, are able to reason about another agent using their own models of that other agent and use this selectively to influence the other agent and agree on a course of action to take, this process can be abstracted into a persuasion-deliberation dialogue. The thesis describes a framework for plan proposals based on a defeasible [2] argumentation scheme and associated critical questions to allow agents to reason about plans and critiques.

The argumentation scheme for plan proposals and associated critical questions appropriate for discussing multi-agent plans. The argumentation scheme representation treats a plan proposal as a *presumptive argument* for a sequence of actions to be executed by a group of agents. The proposal comes with a set of critical questions whose answers may defeat the presumption at various levels and draw attention to potential inconsistencies in the proposal and other alternative ways of reaching the goal. I identified critical questions (Section 3.5) related to the argumentation scheme where each question represents a way to question and/or attack the plan proposal. The scheme has a large number of elements, and consequently the set of critical questions is necessarily large. The large number of questions is necessary to cover the potentially many differing details that make planning such an intricate, fine-grained process.

I present also an argumentation-based protocol called *PDGP* based on the argumentation scheme and critical questions that allows agents to propose plans to each other and evaluate them. In a planning scenario agents require a communication medium with flexibility. In principle, agents should be able to enter into several types of dialogues (negotiation, deliberation persuasion, information seeking, etc.) to agree on a plan [58]. I focus on a persuasion dialogue where agents acquire a role and evaluate the proposal according to their role. Furthermore, a deliberation dialogue could be used to modify a plan at a particular point.

[2]Defeasible reasoning is reasoning where a conclusion reached at one time may be overturned at a later time, for example, if new information is received.

Also a strategy to identify, prioritise and select questions is presented. Choosing an appropriate question in the dialogue becomes an important issue in terms of dialogue and cooperation efficiency. The critical questions are chosen in the context of a dialogue to lead agents to cooperate more effectively issuing as lees questions as possible.

I also implemented a simulated dialogue scenario, described in Chapter 6, in which two conversational agents propose, critique plans and agree on a plan. Agents share their plans and trying to agree on a plan to execute through dialogue. Conflict of course can occur given agents having different views of the world (different beliefs), different ways to achieve the goal, or different preferences with respect to sub-goals. Through the protocol rules agents are able to identify and resolve the conflicts identified.

When selecting questions agents should consider several factors, including the context in which the dialogue develops and the nature of the questions.

1.4 Contribution and Evaluation

The contribution of the thesis is a method that allow agents to select and evaluate plan proposals, more specifically:

- A novel argumentation scheme for plan proposals that allow agents to propose multi-agent plans that builds from an action proposal.
- A novel list of critical questions related to the argumentation scheme for plan proposals where questions are categorised according to the detail in which agents discuss the plan.
- A dialogue game protocol that enables plan proposals to be evaluated, critiqued and modified.
- A methodology to identify critical questions for a plan proposal and a strategy to prioritise and select critical questions in a dialogue about multi-agent plans.
- A computational implementation of the argumentation scheme and critical questions, the dialogue protocol, the strategy to select critical questions and a simulated dialogue between two agents.
- An evaluation of the strategy to select questions using the implementation where strategies are evaluated regarding the number of questions needed to end the dialogue.

1.5 Published Work

The research presented in this thesis has been developed under the supervision of Dr. Katie Atkinson, Professor Peter McBurney and Professor Trevor Bench-Capon and arts of it have been published as follows:

- The argumentation scheme for plan proposals presented in Chapter 3 builds on work undertaken with Peter McBurney, Katie Atkinson and Trevor Bench-Capon and was published as "Arguments over co-operative plans" in the *First International Workshop on the Theory and Applications of Formal Argumentation, TAFA11* in Barcelona, Spain, 2011, [120].

- The Critical Question analysis builds on work undertaken with Peter McBurney and Katie Atkinson in 2010 and published as two technical reports in the Department of Computer Science in the University of Liverpool [116, 121].

- An early version of the *PDGP* dialogue game protocol of Chapter 4 is published as a technical report the Department of Computer Science in the University of Liverpool, 2012 [117].

- The strategy to select critical questions of Chapter 5 builds on work undertaken with Katie Atkinson and Trevor Bench-Capon, an abridged version was published as "Persuasion Strategies for Argumentation about Plans" in the proceedings of the *Fourth International Conference on Computational Models of Argument (COMMA 2012)* in Vienna, Austria, 2012 [119].

- A full version of the strategy to select critical questions that includes the implementation and evaluation presented in chapters 6 and 7 builds on work undertaken with Katie Atkinson, Trevor Bench-Capon and Peter McBurney and was published in the *Journal of Argument and Computation* in March 2013 [118].

1.6 Thesis Outline

The thesis is structured in the following way:

Chapter 2 presents a literature review on the topics that are related to the research presented in this thesis. These topics include: an overview of agent technology and agent communication in Section 2.1, an introduction to Argumentation in AI and dialogue games based on Argumentation in Section 2.2, a review of practical reasoning for autonomous agents in Section 2.3, and approaches to argumentative dialogues for cooperative plans in Section 2.4.

Chapter 3 presents a framework for plan proposals that consists of an argumentation scheme and its associated critical questions together with an analysis on durative actions.

Chapter 4 presents a dialogue game protocol based on the argumentation scheme and critical questions of Chapter 3.

Chapter 5 presents strategies to identify, select and prioritize critical questions in automated dialogues. Different questions become relevant at different times in the dialogue. Here some of the factors that make questions relevant at different times in a dialogue are considered; these can be used to create sensible orderings for questions to be posed in a dialogue.

Chapter 6 presents an implementation of the concepts presented in Chapters 3, 4 and 5. To establish the relevance of our approach I implemented two agents that engage in a dialogue where agents have different views of the world and different preferences and use the strategies from Chapter 5 to select their critical questions.

Chapter 7 presents the evaluation of the experiments. The results show how critical questions implemented in a dialogue game, together with a strategy to choose relevant questions, is beneficial both to the quality and efficiency of the dialogue and the plan which results from it. In contrast to existing work that is largely theoretical, our novel implementation enables us to produce empirical results showing the benefits of our approach in teasing out the points of disagreement to come to an agreement on the best plan.

Chapter 8 presents the thesis conclusions and identifies possible future research paths.

Chapter 2

Literature Review

This chapter presents a survey of research related to the work developed in the thesis. To provide the basic concepts about *Argumentation-based Dialogues about Cooperative Plans*, this chapter presents research related to several disciplines including: Agent Technology, Agent Communications, Interaction Protocols, Practical Reasoning for Autonomous Agents, Argumentation, Computational Models of Argument and Dialogues about Plans.

As discussed in the previous chapter, the main purpose of this thesis is to present theories and mechanisms related to how agents should engage in a dialogue about cooperative plans. In order to put in context the research presented in this thesis, this chapter reviewing the literature is divided into four main sections [1]:

1. ***Section 2.1 Agent Technology and Agent Communication***
   This section presents an overview of the Agent Technology scene focusing on Agent Communication. The section presents agent technology and multi-agent systems and discusses the importance of communication for autonomous agents.

2. ***Section 2.2 Argumentation in Artificial Intelligence***
   This section presents an overview of *Argumentation in Artificial Intelligence* and its importance for enabling agents to reason under uncertainty. Section 2.2.1 presents basic definitions from argumentation theory and continues with research on computational models of argument in Section 2.2.2. Section 2.3.2 presents concepts on Argumentation Schemes and Critical Questions, the basis of our proposal in Chapter 3. Section 2.2.2.1 presents Argumentation Frameworks, Value-based Argumentation Frameworks and Extended Argumentation Frameworks. Finally, Section 2.2.3 presents an overview of Dialogue Games for Agent Argumentation discussing the transition from *Agent Communication Languages* to the *Protocol-based approach* which has become dominant in recent years and has been used in this thesis. In Chapter 4, I will present a protocol that allows persuasion and deliberation based on theories from this section.

[1]This chapter discusses the background research on which the theoretical model developed is based. The background research related to the implementation is presented Chapter 6.

3. ***Section 2.3 Practical Reasoning in Autonomous Agents***
   This section discusses practical reasoning for autonomous agents and an approach to practical reasoning using argumentation schemes for action proposals which is the basis of our plan proposal in Chapter 3.

4. ***Section 2.4 Dialogues and Plans***
   This section discusses the problem of coordinating multiple autonomous agents in relation to planning tasks and the communication issues involved. I present also a survey of existing approaches to overcome some AI planning limitations using argumentative dialogues.

The chapter concludes with Section 2.5 presenting a summary of the key concepts that will be relevant for the rest of the thesis.

2.1 Agent Technology and Communication

An overview of the agent technology in general and in particular agent communication is presented in this section.

2.1.1 Agent Technology

Computing currently is conceptualized as a social activity where a large number of heterogeneous computational nodes are able to communicate through networks. Social networks, e-business platforms, web services platforms, auction and trading websites are all examples. Software applications are no longer isolated and technologies supporting this social trend have been emerging since the nineties. This tendency is very likely to increase in the future with a higher degree of autonomy in software where computational nodes will be able to interact autonomously to reach particular goals [102]. Agent technology thus emerges as a promising paradigm to face the huge task to conceptualize, design and develop computer entities that can meet these requirements to interact in an autonomous and intelligent way, but for this to be accomplished, new characteristics and techniques need to be developed and implemented in computational entities.

The term *software agent* emerged from Artificial Intelligence (AI) research to describe software capable of acting in an autonomous way directed by goals and interacting in open environments [101]. Agents bring together many of the traditional properties of AI programs like pro-activeness, knowledge-level reasoning and goal-directed behaviour with insights gained from distributed software engineering, machine learning and the social sciences. Agent technology has been used as a concept to bring together AI sub-disciplines such as knowledge representation, machine learning, planning, automated reasoning and game theory among others [159]. Wooldridge in [197], defines a software agent as follows:

"*An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives.*"

Agent technologies have been a popular research topic in academia for the last twenty years [92, 102, 199], for example: agent architectures [151], software engineering techniques [101, 162, 197, 200], agent communication [40, 98, 163, 168], agent coordination theories and mechanisms [127, 195, 201] and agent programming languages [17, 31, 47, 87, 126].

Autonomy is probably the most important characteristic wanted for agents, since we want agents to react and pursue their goals without human intervention. Agents are defined as *intelligent*, to refer to their ability to: *communicate* with each other using an expressive communication language; *work together* cooperatively to accomplish complex goals; *act and reason* on their own initiative and use local information and knowledge to manage local resources and *handle requests* from peer-agents [69]. Agent technology is grouped into three categories according to the scale at which it applies [102]:

- The **Agent-level**: involves procedures for individual agent reasoning, planning and learning.
- The **Organization-level**: covers issues related to agent societies; topics include: organizational structure, trust, norms and obligations, and self-organization in open agent societies.
- The **Interaction-level**: concerns communication between agents. Topics include: communication languages, interaction protocols and resource allocation mechanisms. The goal is to develop computational theories and technologies for agent interaction, communication and decision making based on work from other disciplines (economics, political science, philosophy) that have studied similar problems.

More specifically, the *interaction level* can be divided into three main topics: Coordination, Negotiation and Communication [102]:

- **Cooperation** aims to ensure that actions of independent agents are coherent with respect to a common goal, in other words to align goals across agents with the use of commitments or contracts.
- **Negotiation** is concerned with the establishment of mechanisms to divide scarce resources between agents in a way acceptable to all parties, with each individual party aiming to maximize its share.
- **Communication** aims to provide standards and effective solutions for agents to share their goals, plans, beliefs, and preferences. In this context, the ability of agents to communicate is essential. Communication is important because in open

and dynamic environments where resources and information could be spread across different locations, agents need to cooperate, negotiate and interact efficiently to reach their goals.

An important issue of the *interaction level* is how to design, develop and implement mechanisms to coordinate actions automatically between agents. Conflicts may arise because of conflicting goals and preferences and so mechanisms to negotiate are also important. Recent approaches drawn from economics, social choice theory and argumentation (discussed in the next section) provide means to enable richer interactions between agents.

2.1.2 Multi-agent Systems

Particularly complex or unpredictable problems can only be reasonably addressed by developing modular components or agents that are specialists at solving a particular problem aspect [200]. The emergence of the World Wide Web over the last twenty years has been probably the most important driving force for agent technology. Agent-based computing is a multidisciplinary field which is rooted in distributed Artificial Intelligence and distributed object technologies. Both benefits (vast resources available, technologies for remote distribution of information) and difficulties (information gathering interoperability) present in the Web have contributed to the progress of agent technology. Distributed Artificial Intelligence evolved naturally into the field of Multi-Agent Systems (MAS).

MAS are systems that consists of a number of agents which interact with one another. Typically agents will be acting on behalf of users or owners with different goals or motivations in a cooperative environment. Agents therefore, should be designed to fulfil a specific purpose. If an agent performs too many tasks, the complexity of development and maintenance increases beyond acceptable level. If agents must perform several tasks, we can either increase their complexity, or distribute these tasks among specialized agents and make them work co-operatively and communicate effectively. Therefore the need of a formal and standard way to allow communication between agents arises.

2.1.3 Agent Communication

Communication among software agents is an essential property of agency [199]. Communication in this context concerns more than the act of exchanging messages in a certain format. The meaning of utterances, semantic verification and development of formal theories for language and protocol properties are necessary for a wider adoption of research findings in the communication area.

Agent communication should allow agents to exchange information with other agents despite differences in hardware platforms, operating systems, programming languages and representation and reasoning systems. More importantly communication in computing has to be seen as knowledge sharing instead of just message interchange. That is why research in agent communication aims to create languages and protocols as standards to be used in a wide range of agent-oriented implementations despite differences in their technologies.

An **A**gent **C**ommunication **L**anguage (*ACL*) is the medium through which the content of messages (requests, queries, assertions) between agents is communicated. Several technologies have been developed to achieve the goal of exchange of information between applications, for example: Remote Method Invocation (RMI) [55] (a Java application programming interface for performing the object equivalent of remote procedure calls) and CORBA [2] [182] (a middle-ware software that allows programmers to make program calls from one computer to another via a network). Distributed object technologies like *CORBA* [182], have provided a solid infrastructure to handle low-level inter-operation of heterogeneous distributed components.

What distinguishes *ACLs* from such efforts are the objects of discourse and their semantic complexity. Agents also are autonomous unlike the objects in RMI and CORBA. Furthermore, agents should be able to exchange not just single utterances but more complex objects such as plans, goals or shared-experiences [98].

There are two basic approaches to designing an *ACL*. The first is procedural where communication is based on executable content. This approach presents limitations because executable content is difficult to control, coordinate and merge, and all these characteristics are highly desirable for agent communication languages. The second approach is declarative where communication is based on illocutionary acts or performatives (speech acts), such as requesting or commanding.

An illocutionary act [14] is a speech act made using an utterance that consists of the delivery of the propositional content of the utterance with a particular illocutionary force whereby the speaker asserts, suggests, demands, promises, vows, etc. The theory of speech acts was first introduced by philosopher John Austin [14] and greatly developed by John Searle [161].

The theory identifies performative verbs which correspond to speech acts. This theory of speech acts has been adapted for expressing interactions between agents in ACLs. The theory of rational action presented in [44] connects the theory of speech acts with rational processes. The rational action theory conceptualizes speech acts as intended actions performed by agents to satisfy their intentions.

One of the first and most important efforts to create an *ACL* was the Knowledge Query Manipulation Language (***KQML***) developed in the 1990s by the US-based DARPA-funded Knowledge Sharing Effort [137] (*KSE*). The KSE group started defining propositional attitudes which are the basic concept for representing formally a communicative act. From there the *KSE* group identified key points to come up with a common-language (syntactic translation, semantic content and communication of complex attitudes).

[2]The Common Object Request Broker Architecture (CORBA) is a standard defined by the Object Management Group that enables software components written in multiple computer languages and running on multiple computers to work together.

Two of the most important grounds on which *KQML* was criticized were that its semantics were never rigorously defined [3] and that it lacks performatives by which agents make commitments to each another. Furthermore, whether an agent complies to the sincerity condition is impossible to verify in practice [198], since the use of propositional attitudes in the definition of the acts made it impossible for observers to determine if the performatives were being used properly and sincerely.

Criticism of the *KQML* language led to the development of improved languages like ***FIPA-ACL*** [4] [70]. *FIPA* defines specifications that deal mainly with: agent communication, agent transport, agent management, abstract architecture and applications, *agent communication* being the core category of the model. Specifically, *FIPA-ACL* defines specifications that deal with:

- *ACL* messages (i.e. structure, XML representation, etc.).
- Speech act theory-based communicative acts (the *FIPA* Communicative Act Library Specification [70] defines twenty two performatives to communicate messages).
- Operational semantics using speech act theory.
- Content language specifications.

The *FIPA-ACL* is similar to *KQML* in the sense that it defines an 'outer' language for messages. The semantics of *FIPA-ACL* were given with respect to the *SL* formal language that allows the representation of beliefs, desires and intentions as well as the actions that agents perform. Each *ACL* message is mapped to a formula that defines a constraint (feasibility condition) that the sender of the message must satisfy if it is to be considered as conforming to the *FIPA-ACL* standard. The key weakness of *FIPA-ACL* from the argumentative point of view is its limited support to change their position or self-transformation by participants [112], since its designers did not seek to embody a deliberative democratic view but a rational-choice view of agent society. Another critique in [138], discusses the *sincerity condition* in some *FIPA-ACL* locutions as a disadvantage. The sincerity condition requires that an agent believe whatever it informs another agent. In self-interested agents with private goals sincerity may not be obligatory. Note that the semantics of *FIPA*, like *KQML*, are based on unobservable propositional attitudes.

In a review paper in 2000 [168], Tadiou et al. state that a generalized *communication framework* should be characterized by the following set of principles:

1. *Heterogeneity principle:* the meaning of messages should be context independent and reflect a global perspective.

[3]Although *KQML* has been given some formal semantics by the original authors in [97] it was not considered comprehensive by its critics.

[4]The Foundation for Intelligent Physical Agents (FIPA) is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies.

2. *Cooperation and Coordination principle:* appropriate interaction protocols for a given task should exist.

3. *Separation principle:* message content, structure and transport mechanism should be handled separately.

4. *Interoperability principle:* agents should be able to inter-operate despite different architectures or programming languages.

5. *Transparency principle:* complexity of the underlying *ACL* should be shielded.

6. *Extensibility principle:* new communicative acts and interaction protocols should work with existing ones.

7. *Performance principle:* an *ACL* should be efficient, reliable, safe and secure.

I continue the discussion of Agent Communication in section 2.2.3 where I present the transition from *ACLs* to conversation policies to the most recent approach using specific dialogue protocols. The next section presents concepts of Argumentation and its relation to Artificial Intelligence and research related to computational models of argument.

2.2 Argumentation in Artificial Intelligence

Argumentation is concerned primarily with reaching conclusions through reasoning with incomplete or inconsistent information, goals or preferences [180]. Argumentation in Artificial Intelligence is mainly used to design reasoning mechanisms for multi-agent systems under uncertainty. The characteristics of argumentation in the AI context such as: the definition of argument components and their interaction, the identification of rules describing argumentation processes and the use of semantics to identify legitimate systems, make argumentation particularly suitable to design reasoning mechanisms for multi-agent systems [23].

An *argumentation system* is a collection of "defeasible proofs" (arguments), that is partially ordered by a relation expressing the difference in conclusive force [183]. The development of argumentation systems has become an increasingly important research topic in Artificial Intelligence, and includes research activities such as developing theoretical models, prototype implementations, and application studies.

In particular, argumentation provides a general framework for inference and decision making in the presence of inconsistent, uncertain or incomplete information or conflicting goals or preferences. A sound justification may be enough to resolve a discussion. Argumentation has a number of applications in Artificial Intelligence such as decision-making under uncertainty, the semantics of logic programming and defeasible reasoning.

In essence, argumentation provides Artificial Intelligence mechanisms to reason about uncertain or incomplete information and justify claims. The problems of understanding argumentation and its role in human reasoning have been addressed by many researchers in different fields including philosophy, logic, legal theory and Artificial Intelligence.

Some of the research topics in argumentation related to Artificial Intelligence are [23]:

- The definition of the component parts of an argument: arguments in logical terms e.g.[27, 144] and the analysis and formalization argumentation schemes and critical questions, e.g. [11, 154, 186]
- Argumentation frameworks: the semantics and fundamental properties of abstract argumentation systems e.g., [20, 56, 123]
- Logic-based deductive Argumentation e.g.,[26]
- Assumption-based Frameworks e.g.,[30, 57]
- Argument and Dialogue: dialogue games, abstract dialogues set in argumentation frameworks, dialogue protocols e.g., [108, 111, 142, 153]
- Computational properties of argument systems e.g., [15, 59, 60, 183]

In the following subsections I present an overview of argumentation theories, and argumentation related to Artificial Intelligence.

2.2.1 Argumentation Theory

Argumentation is a process by which a human attempts to convince another of the truth or falsity of some state of affairs by generating arguments that others cannot object to [180] or convince another of the the desirability of an action e.g. [11]. Argumentation is useful in scenarios where proof cannot be used, such as in domains where information is uncertain or incomplete, and enables reasoning to proceed in a non-monotonic manner or where agents may legitimately have different goals and preferences [23]. But argumentation is also an activity of reason: when people put forward their arguments in argumentation they place their considerations within the realm of reason, thus argumentation is also used to convince others about the desirability of actions.

Elements from logic and formal deductive reasoning have provided basis for modelling argumentation in AI [42]. In [23], Bench-Capon and Dunne present issues when considering argumentation in computational models as following:

- Defining the component part of an argument and their interaction.
- Identifying rules and protocols describing argumentation processes.
- Distinguishing legitimate from invalid arguments.
- Determining conditions under which further discussion is redundant.

In [176], Toumlin identifies key components that define the structure of an argument (see Figure 2.1). The example presented is not exhaustive, other theories of argumentation exists e.g. [185, 196].[5]

Argument components as in [176] are:

- Facts: information that is specific to a given context.
- Warrants: relates the facts to qualified claims
- Backing: justification of a warrant.
- Rebuttal: exceptions of a warrant.
- Qualified Claim: conclusion drawn if the warrant holds, and the confidence in that conclusion (possibly, typically, certainly)

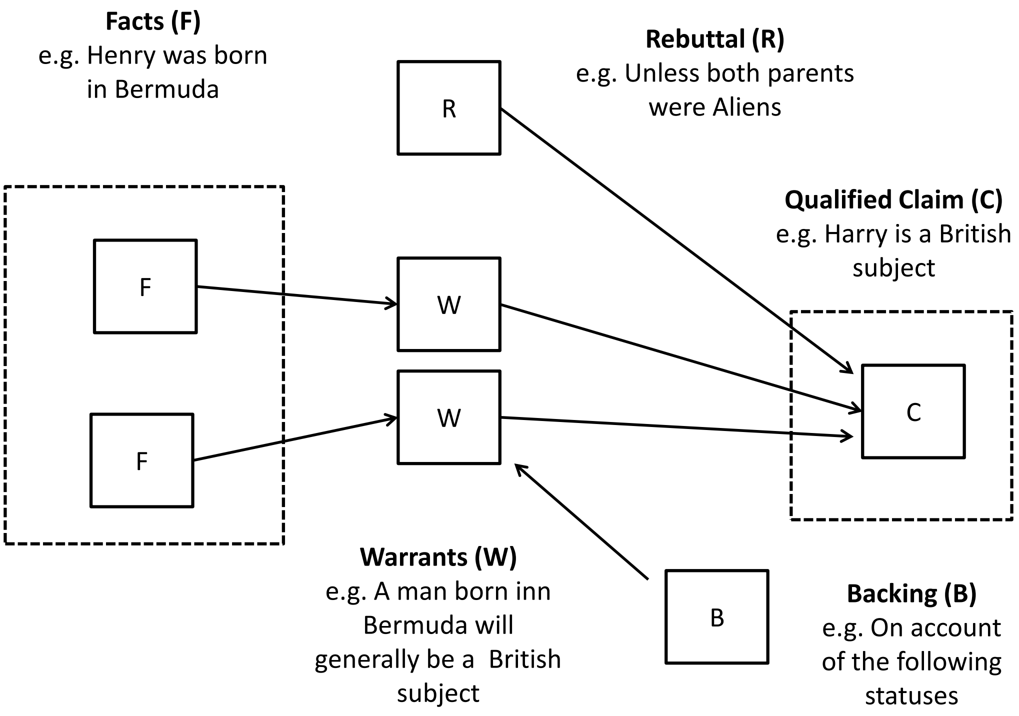

Facts (F)
e.g. Henry was born in Bermuda
Rebuttal (R)
e.g. Unless both parents were Aliens
R
Qualified Claim (C)
e.g. Harry is a British subject
F
W
W
F
C
Warrants (W)
e.g. A man born inn Bermuda will generally be a British subject
B
Backing (B)
e.g. On account of the following statuses

Figure 2.1: Graphical Representation of an Argument. Adapted from an example in [176].

Toumlin's layout of arguments provides important layout to understand argument structure but lacks a comprehensive account on the logical underpinnings of argument and does not address how to construct arguments. I present now formal argumentation definitions taken from [27] that address arguments from the logical perspective.

[5]The diagram structure introduced by Wigmore [196] is used primarily for the analysys of legal arguments.

An argument is a set of appropriate formulae Φ, that can be used to prove a claim α, together with that claim.

Definition 1. An **argument** A is a pair $< \Phi, \alpha >$ such that:

1. $\Phi \nvdash \bot$.
2. $\Phi \vdash \alpha$.
3. Φ is a minimal subset of Δ satisfying 2 where Δ is a large repository of information from which arguments can be constructed.

If $A =< \Phi, \alpha >$ is an argument, we say that A is an argument for α and we also say that Φ is a support for α. We call α the claim or the consequent of the argument, and Φ the support.

Definition 2. A **defeater** for an argument $< \Phi, \alpha >$ is an argument $< \Psi, \beta >$ such that $\beta \vdash \neg(\phi_1 \wedge ... \wedge \phi_n)$ for some $\{\phi_1, ..., \phi_n\} \subseteq \Phi$.

Definition 3. An **undercut** for an argument $< \Phi, \alpha >$ is an argument $< \Psi, \neg(\phi_1 \wedge ... \wedge \phi_n) >$ where $\{\phi_1, ..., \phi_n\} \subseteq \Phi$ i.e. some arguments directly oppose the support of others.

Definition 4. An argument $< \Psi, \beta >$ is a **rebuttal** for an argument $< \Phi, \alpha >$ iff $\beta \leftrightarrow \neg\alpha$ is a tautology i.e. when two arguments have opposite claims.

An alternative recursive definition of arguments in terms of trees of sub-arguments is given in [144]. In recent years there has been a rapid growth of interest in the use of models of argumentation for modelling agent reasoning (agents can use argumentation-based models as a means of performing their own internal reasoning through the evaluation of arguments [56]) and dialogue.

2.2.2 Computational Models of Argument

2.2.2.1 Argumentation Frameworks

Abstract Argumentation focuses on arguments with no internal structure; other approaches provide a formal analysis of the internal structure of arguments, e.g. [27, 144]. This section presents the *Argumentation Framework* (*AF*) of Dung [56] which is considered the basis to study abstract argumentation[6] *AF*s allow us to analyze a system of conflicting arguments in which arguments are abstracted and the role of arguments is determined solely by a single relation to other arguments normally called “defeat”. The idea of reasoning with *AFs* is to identify which arguments can be accepted in relation to the other arguments in the framework, given this defeat relation. In [56], Dung states:

[6]From now I use the terms Argumentation System and Argumentation Frameworks interchangeably.

"The understanding of the structure and acceptability of arguments is essential for a computer system to be able to engage in exchanges of arguments".

In [56], Dung defines an AF as a finite set of arguments, and a binary relation between pairs of these arguments called an *attack*. Although referred as "attack" in [56] this relation is more accurately referred as "defeat", as in much of the later work on AFs attacks can be successful (i.e. defeats) or unsuccessful. Arguments are viewed as atomic and no concern is given to the internal structure of the arguments or the nature of the attack relation. In [56] all arguments are of equal strength and every attack succeeds. Thus an argument can only be defended by defeating all its attackers. Argumentation Frameworks are often modelled as directed graphs showing which arguments attack one another.

A way to evaluate the status of an argument is to consider whether or not it can be defended against attack from other arguments by a subset of arguments. Sets of arguments can then be evaluated to determine whether or not they are acceptable, according to different types of semantics. AFs abstract many logical systems that have been used to formalize common-sense reasoning[7] or to give a meaning to logic programs. AFs provide a unifying tool for the study of several aspects of logical systems, notably their semantics [36, 54].

Formally, AFs are defined as a finite set of arguments AR, and a relation between pairs of arguments called an attack, $AF =< AR, attacks >$. An attack from A to B means that argument A defeats argument B. Figures 2.2 and 2.3 presents examples of two Argumentation frameworks as graphs.

The semantics of AFs are given in terms of subsets of AR, called extensions, where each extension corresponds to a possible interpretation of the debate. Argument-based semantics can be seen as based on argument extensions or sets of arguments with special properties. Admissible sets provide the basic context for examining ideas of "maximally consistent sets of beliefs". By applying some particular semantic rules on an Argumentation Framework, the output is one or more sets of arguments that survive the competition between all arguments. These extensions represent a defensible point of view or a position that supports a statement or an argument. To provide definitions of the extensions first I provide some definitions:

Definition 1: A set of arguments S is ***conflict free*** if there are no arguments A and B in S such that A attacks B. Formally:

$\neg(\exists x)(\exists y)((x \in S)\ \&\ (y \in S)\ \&\ attacks(x, y))$

Definition 2: An argument $A \in AR$ is ***acceptable*** with respect to a set of arguments S, iff each attacker of A is attacked by any argument in S. Formally:

$acceptable(x, S) \text{ if } \forall x((x \in AR)\ \&\ (attacks(x, A)) \implies (\exists y)(y \in S)\ \&\ attacks(y, x)$

[7]The stable marriage problem [115] for example could be modelled and solved using Argumentation Framework semantics.

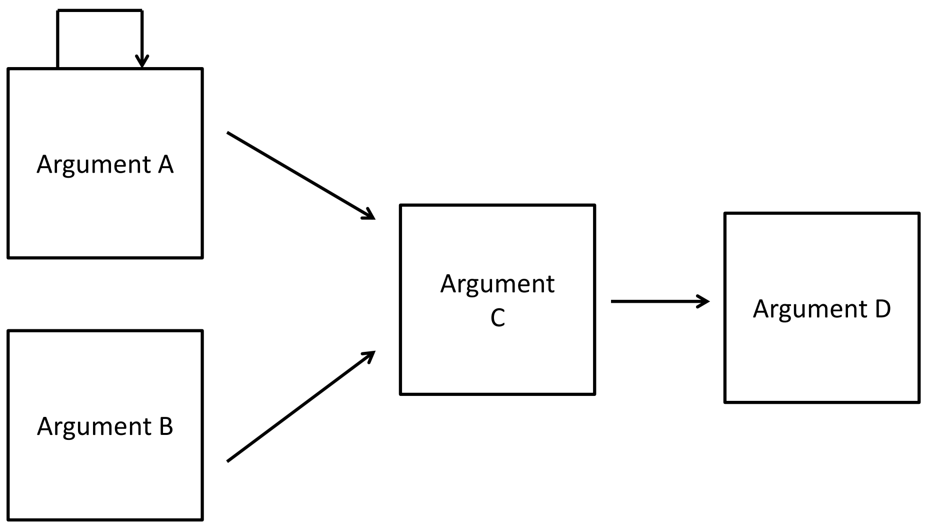


FIGURE 2.2: An Argumentation Framework with four arguments. Argument A is a self-attacking argument.

Definition 3: An ***admissible*** set of arguments is a conflict free set in which every argument of S is acceptable with relation to S. Formally:

$(\forall x)(x \in S) \implies acceptable(x, S)$

Definition 4: ***Reinstatement*** is defined as the act to restore a previous condition or position. Defeated arguments should be regarded as justified, as long as the arguments defeating them are themselves defeated [146]. The idea of *reinstatement* derives from [56], through the concept of *acceptability.* Let's suppose, that a theory allows for the construction of three arguments A, B and C subject to the following defeat relations: B *defeats* A, and C *defeats* B. In such a case, , the argument C should be thought of as reinstating A, by defeating the only argument that defeats it, so that A itself is to be regarded as justified.

A ***Grounded Extension*** is the least fixed point of an acceptable set of arguments. Grounded semantics induce a unique extension of admissible arguments for each AF. The grounded extension is unique and there exists always some grounded extension, but it might be the empty set. For example, the grounded extension for the Argumentation Framework in Figure 2.2 is $\{B, D\}$, argument A defeats itself and argument C is defeated by argument B. For the AF in Figure 2.3 the grounded extension is the empty set, since no argument is non-attacked.

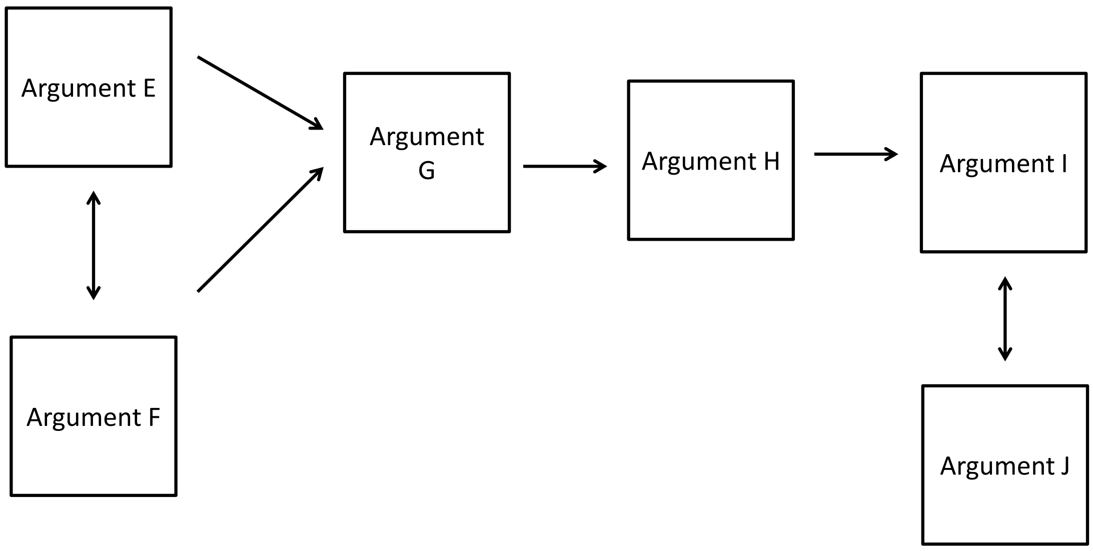


Figure 2.3: An Argumentation Framework with six arguments. Arguments E and F attack each other.

A ***Preferred Extension*** of an argument system is a maximal (with respect to set inclusion) admissible set of arguments in AF. This means that a preferred extension is always admissible, conflict free and all the arguments in it are acceptable. In a preferred extension a set of arguments S defends every argument it contains, and is always the largest possible set of arguments that conforms to these properties, thus an argument system may have more than one preferred extension. Different preferred extensions arise from even cycles in the corresponding graph, which necessitate a choice. An argument which is acceptable in all preferred extensions is said to be "sceptically" acceptable. One which is acceptable in at least one, but not all, is said to be "credulously" acceptable.

An argument is defensible if it appears in at least one preferred extension, and indefensible otherwise. The preferred extension for the AF in Figure 2.2 is $\{B, D\}$ and for the AF in Figure 2.3 extensions $\{E, H, J\}$ when E defeats F and $\{F, H, J\}$ when F defeats E. Note that because I is defeated by H, no choice between I and J is needed. Some interesting properties of preferred semantics are (taken from [184]):

– Every AF possess at least one preferred extension.

– The preferred extension might be the empty set.

– Multiple extensions give rise to richer acceptance classes, but this creates complications with interpretation.
– Every admissible set is contained in some preferred extension.
– An AF without cycles has exactly one preferred extension.
– Every grounded extension is a preferred extension.

A ***Stable Extension*** of an AF is defined as a conflict-free set of arguments S that attacks every argument which does not belong to S. An stable extension represents a set of arguments that attacks every other argument outside of it so that the point of view is stable providing counterarguments for every attack on their set. The stable extension for the AF in Figure 2.4 is $\{A, B, D\}$.

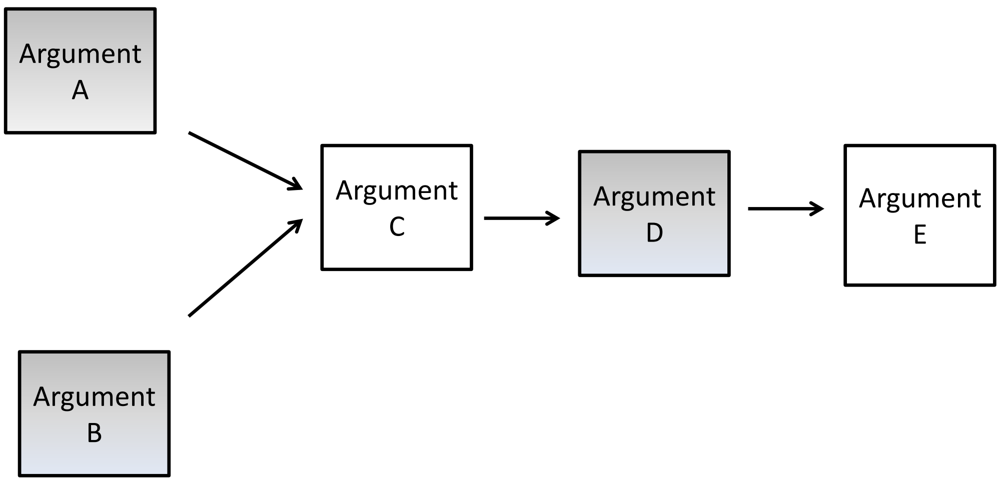


FIGURE 2.4: An Argumentation Framework with five arguments. The *stable extension* is arguments $\{A, B, D\}$.

An admissible set S of arguments is called a ***complete extension*** iff each argument, which is acceptable with respect to S, belongs to S. Complete extensions capture the kind of confident agent who believes in everything it can defend.

2.2.2.2 Value-based Argumentation Frameworks

There are several ways to distinguish "attacks" from "successful attacks" (i.e. defeats) based on preferences over arguments. An early attempt was the definition of preference based frameworks in [4]. This thesis will mainly use Value-based Argumentation Frameworks (*VAFs*) as presented in [19, 20]. More recently, an account generalizing all these approaches has been given in [124].

VAFs provide an interpretation of multiple preferred extensions in a single argument system. Dung's formalism is then extended to provide a semantics for distinguishing and choosing between consistent but incompatible belief sets through the use of *argument values* [19]. These values are interpreted as the social values promoted by the argument, and the strength of an argument for an *audience* depends on how that audience orders these values.

VAFs make defeat dependent on the relative importance of the values the arguments advance or protect, an attack fails if the value of the attacked argument is preferred to that of the attacker. The process of determining if an argument is acceptable in *VAFs* is different, since the values of the arguments need to be considered. *VAFs* with two distinct values resolve the problem of cycles in *AFs* because the rule to determine if an argument defeats another is based on the value that both arguments have. If a cycle contains two or more distinct values, it has a unique, non-empty preferred extension.

The idea is to re-introduce an element which has been abstracted away in standard *AFs*, and which can be used to ground a rational choice between alternatives which are equally tenable from the more abstract point of view. In a *VAF* it may be possible to force rational acceptance of particular arguments irrespective of how the values are ranked [20]. A *VAF* is defined by a triple:

$\langle \mathcal{H}(\mathcal{X}, \mathcal{A}), \mathcal{V}, n \rangle$
where:

- $\mathcal{H}(\mathcal{X}, \mathcal{A})$ is an argument system, in which $\mathcal{X}$ is a finite set of arguments and $\mathcal{A} \subset \mathcal{X} \times \mathcal{X}$ is the attack relationship for $\mathcal{H}$.
- $\mathcal{V} = \{v_1, v_2, ...v_k\}$ a set of k values, and
- $n : \mathcal{X} \rightarrow \mathcal{V}$ a mapping that associates a value $n(x) \in \mathcal{V}$
- with each argument $x \in \mathcal{X}$.

We say that an argument $x \in \mathcal{X}$ relates to value v_n if accepting x promotes or defends v: the value in question is given by n. In a simple *AF* it is assumed that an attack always succeeds and so, the purpose of extending the argumentation framework is to distinguish between one argument attacking another, and that attack succeeding. So, an attack succeeds if both arguments relate to the same value, or if no preference between the values has been defined. Note that if $\mathcal{V}$ contains a single value, the *VAF* becomes a standard *AF*. In *AFs* odd length cycles in a single value represent paradoxes and even length cycles are dilemmas giving rise to two preferred extensions; in *VAFs* they are resolved for a given *audience* according to its value preference.

An *audience*[8] for a *VAF* $\langle\mathcal{H}(\mathcal{X},\mathcal{A}),\mathcal{V},n\rangle$, is:

- a binary relation $\mathcal{R} \subset \mathcal{V} \times \mathcal{V}$
- whose (irreflexive) transitive closure $\mathcal{R}^*$,
- is asymmetric, i.e. at most one of $< v, v' >$, $< v', v >$ are members of $\mathcal{R}^*$
- for any distinct $v, v' \in \mathcal{V}$,
- we say that v_i is preferred to v_j in the audience $\mathcal{R}$, denoted $v_i \succ_{\mathcal{R}} v_j$, if $< v_i, v_j > \in \mathcal{R}^*$ and
- a specific audience α is compatible with $\mathcal{R}$ if α is a total ordering of $\mathcal{V}$.

VAFs replace the concept of attacks in [56] and defines *successful attacks* with respect to an audience $\mathcal{R}$ if: $< x, y > \in \mathcal{A}$ and it is not the case that $n(y) \succ \mathcal{R}n(x)$, in that way, the concepts "argument acceptable to a set", "conflict free", "admissible set", "preferred extension and "stable extension" are also applicable to *VAFs* with respect to audiences [23]. In *VAFs*, the relative ordering of different values promoted by distinct audiences results in arguments falling into one of three categories [21]:

C1. Arguments, x, that are in the preferred extension $P(\langle\mathcal{H}(\mathcal{X},\mathcal{A}),\mathcal{V},n\rangle,\alpha)$ for some specific audiences compatible with $\mathcal{R}$ but not all. Such arguments being called *subjectively* acceptable with respect to $\mathcal{R}$.

C2. Arguments, x, that are in the preferred extension $P(\langle\mathcal{H}(\mathcal{X},\mathcal{A}),\mathcal{V},n\rangle,\alpha)$ for every specific audience compatible with $\mathcal{R}$. Such arguments being called *objectively* acceptable with respect to $\mathcal{R}$.

C3. Arguments, x, that are not in any preferred extension $P(\langle\mathcal{H}(\mathcal{X},\mathcal{A}),\mathcal{V},n\rangle,\alpha)$ no matter which specific audience, α, compatible with $\mathcal{R}$ is used. Such arguments being called *indefensible* with respect to $\mathcal{R}$.

An example of a *VAF* is presented in Figure 2.5 where the preferred extension for the audience corresponding to the ordering $v_1 \succ v_2 \succ v_3$ is $\{A, D, F\}$, the preferred extension for the audience $v_2 \succ v_3 \succ v_1$ is $\{A, D, E\}$ and the preferred extension for the audience $v_3 \succ v_2 \succ v_1$ is $\{B, D, F\}$. We can say that $\{D\}$ is objectively acceptable, $\{A, B, E, F\}$ are subjectively acceptable while $\{C\}$ is indefensible.

[8]Audiences are distinguished in terms of values as in [20] also accommodating differences in beliefs.

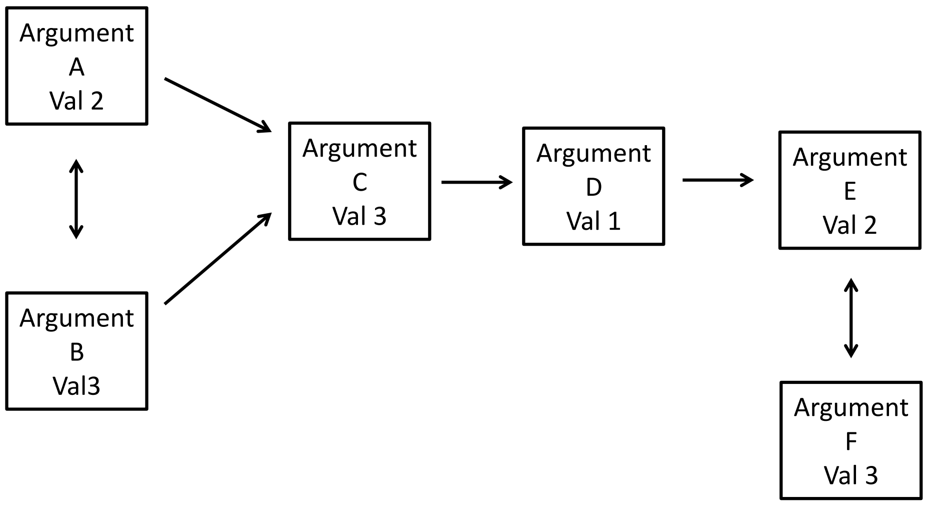


Figure 2.5: An Example of a Value-based Argumentation Framework

2.2.2.3 Extended Argumentation Frameworks

In [123], Modgil introduces the concept of Extended Argumentation Frameworks (*EAFs*). An *EAF* extends Dung's framework to allow meta-level argumentation about preferences. Preferences are expressed using a second attack relation that is characterized by attacking attacks between the arguments that are the subject of the preference claims. In this approach the acceptability of attacks is extended based on Dung's semantics. In this way argumentation about preferences can be modelled in a richer and more expressive way with an extended framework.

An *EAF* includes a second attack relation $\mathcal{D}$ that ranges from arguments X to attacks $(Y, Z) \in \mathcal{R}$, where $\mathcal{R}$ is the standard binary attack relation in a Dung framework. If X attacks (Y, Z) then X expresses that Z is preferred to Y. If X' attacks (Z, Y), then X' expresses that Y is preferred to Z. *EAFs* then are required to conform to the constraint that any such arguments expressing contradictory preferences must attack each other (i.e., $(X, X'), (X', X) \in \mathcal{R}$). Figure 2.6 presents the graph of the *EAF* described. Formally, an *EAF* is a tuple $(Args, \mathcal{R}, \mathcal{D})$ such that:

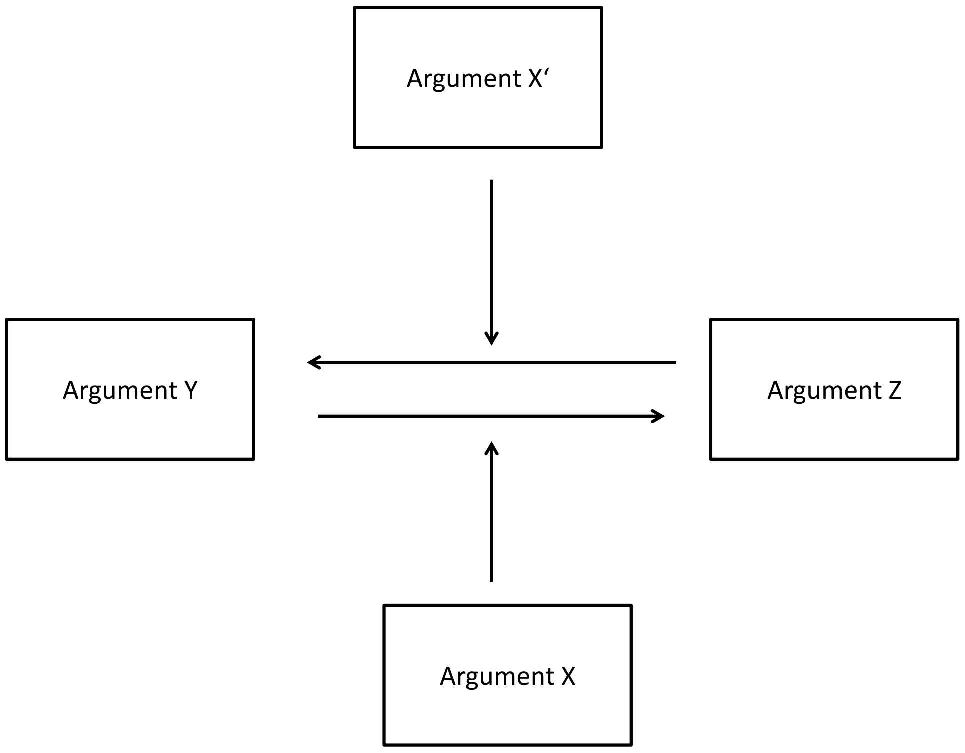


FIGURE 2.6: An Extended Argumentation Frameworks where preferences are represented by a second attack relation. X' expresses that Y is preferred to Z because X' attacks (Z, Y).

- $Args$ is a set of arguments
- $\mathcal{R}$ is a standard binary attack relation in a Dung framework and $\mathcal{R} \subseteq Args \times Args$,
- $\mathcal{D} \subseteq Args \times \mathcal{R}$.
- so, if $(X, (Y, Z)), (X', (Z, Y)) \in \mathcal{D}$ then $(X, X'), (X', X) \in \mathcal{R}$

All the semantics used in *AFs* can be used in *EAFs*. Modelling *VAFs* in *EAFs* is described in [124].

2.2.3 Agent Communication using Argumentation Principles

In Section 2.1.3, I discussed the use of *ACLs* as a mechanism agents use to communicate. In this subsection I present other mechanisms agents use to communicate namely: *Conversation Policies* and, in more detail, *Dialogue Games*.

Table 2.1 summarizes tendencies in agent communication from relevant *ACLs* survey papers. With the use of the table we can see how argumentation-based dialogues based on commitments gained relevance as a popular mechanism for agents to communicate.

TABLE 2.1: Summary of Agent Communication Surveys.

Survey	**Tendency**	**Year**
Labrou et al. [98]	Definition of basic ontologies for communication. Standardization of basic standard protocols.	1999
Tadiou et al. [168]	Shared ontologies and negotiation protocols.	2000
Singh [163]	Protocols with roles that define social commitments.	2002
Chaib-Draa and Dignum [40]	Social Commitments in dialogue Conversation policies Dialogue Games	2002
Parsons and McBurney [132]	Argumentation-based Dialogues	2003
McBurney and Parsons [111]	Dialogue Games for Agent Argumentation	2009

Agent communication interoperability is thought to rest on three main characteristics [78]:

1. Agents able to access a set of shared structural interoperability e.g. infrastructure services for registration, reliable message delivery, agent naming, etc.

2. Agents sharing a common content ontology, truth theory (i.e., there must be logical interoperability); and

3. Agents agree on the syntax and semantics of a common agent communication language.

As a practical matter, agents must also agree on the range of possible sequences and contents of messages when they are interpreted in the context of larger goal directed inter-agent dialogues, or conversations.

These issues rise to a significant ambiguity problem for agents that need to interact using a powerful ACL, which in [78] is called the Basic Problem: "Modern ACLs, especially those based on logic, are frequently powerful enough to encompass several different semantically coherent ways to achieve the same communicative goal, and inversely, also powerful enough to achieve several different communicative goals with the same ACL message".

As I discussed in Section 2.1.3 *KQML* and *FIPA-ACL* were criticized on various grounds. Essentially these languages have been designed for a very general purpose. Given this characteristic agents participating in conversations had too many choices of what to say causing a state-explosion problem, and the definitions had to be given in a very general manner to be applicable to every context in which they might be used.

From a practical point of view, the semantics of communicative acts is so rich that it is far too complex to determine the possible answers by just inferring others mental states, (there are too many semantically coherent dialogue continuations), furthermore agents must interpret context,of the dialogue its relevance, etc [104]

And from a theoretical point a view, mentalistic approaches assume agents mental states are verifiable and agents are sincere. Both hypotheses are problematic, especially in an open environment. The semantic conformance testing issue presented in [198], states that verifying that an agent respects the semantics of an agent communication language is in principle a verification problem focused on the specifications presented in the feasibility preconditions of utterances. But to solve this verification problem when the semantics involve internal propositional attitudes we need to access the mental states of other agents and it is not clear how this might be done. Even if we could access agents' internal specifications we cannot completely verify conditions. This is the main reason not to use private semantics such as the ones in *KQML* and *FIPA-ACL*. Two alternative approaches to the mental model of agency as a basis for communicative acts have been suggested:

- Conversation Policies e.g. [25, 78, 164].
- Semantics based on Dialogue Games and Social Commitments e.g. [91, 104, 111, 142].

2.2.3.1 Conversation Policies

A *conversation policy* is a pre-planned declarative specification that govern communications between software agents using an agent communication language [78] offering the flexibility expected for MAS to support emergent conversations. Agents engaged in dialogue implement decision procedures that allow an agent to select locutions according to its intentions. These decision procedures must take into account the context of prior locutions in the dialogue which could simplify the computational complexity of selecting the message [40]. Dialogues are then much more viable because agents engaging in pre-planned conversations reduce the search space of possible agent responses while still being consistent with a semantics [104].

The concept of conversation policies is based on the fact that the use of language by an agent is no different from any other action that an agent might take [78]. Thus conversation policies use the theoretical framework of speech acts (and their intended perlocutionary effects) so every agent message is driven by a strategy to achieve the agents current goals.

The process of planning and goal satisfaction still may be a completely different process in for a particular agent, so an agent may be executing some pre-built procedure in the context of the dialogue and still performing an explicit deliberation process.

Examples of conversation policies that refer to common agent interactions and sequencing constraints (which make up the traditional subject matter of a conversation policies) are: dialogue termination, synchrony (turn-taking, concurrency of messaging, interruption possibilities), uptake acknowledgment, exception handling, pragmatics (agreement on the preferred way to express a given communicative act), sequence policies, conversation management policies, specific goal achievement policies, request for action policies, etc.

Form a high level perspective a request for action conversation policy could in the following way [104]:

- Conversation begins in the initial state by the request from speaker S.
- The dialogue can successfully be followed by:
    1. the promise to realize the requested action,
    2. come into a negotiation cycle with a counter-proposal
    3. fail.
- at state (1), the addressee will signal that the task has been achieved, (leading to the final state
- or eventually decide to renege

Two main issues related to the specification of conversation policies are: flexibility and specification [104]. Flexibility refers to the between this normative aspect and the flexibility expected in most multi-agent communications Specification require ad hoc formalism to take all the profit from the specification, and formally verifying some expected properties of the model.

2.2.3.2 Social Commitments

Social commitments are commitments that bind a speaker to a community defining clearly a distinction between the "creditor" and the "debtor" of the commitment [104]. This notion of commitment is clearly different from that of belief or intention and the private states of the agents. The essence of taking a social commitment perspective is that agents state publicly their beliefs and intentions, so that future utterances and actions may be judged for consistency against these statements. This is in contrast to the private semantics of *KQML*, which rely on internal propositional attitudes of the participants.

The notion of *commitment* in early dialogue game literature [84] assumes participants incur in a commitment as a *dialectical obligation*n that may be related to the true beliefs of the participant.

Amongst social commitments, a classical distinction is established between propositional or action commitments. A "propositional commitment" binds an agent A with a proposition p towards the audience B, while an "action commitment" binds an action of A towards B. Before concluding that A *believes* what he says, we can express that A is *committed* to the audience, and there are consequences related to such a commitment. In particular, the audience will certainly penalize A if he makes an subsequent statement contradictory to his claim (e.g. the audience might consider A as a non-credible agent).

In Walton and Krabbe [186], this distinction is not relevant, by simply uttering a proposition p , A is committed in a way that constrains its subsequent actions and depending upon context. The notion of commitment thus is broader and can include intended actions. An agent then may then become committed to a number of things, e.g. holding that p, defending that p , not denying that p, arguing that p, etc. Both commitment notions have been used in the agent communication literature [24, 104, 106]. More concretely, a social agency model (in contrast with a mental agency model) provides [104]:

- A high design and execution autonomy.
- A high coverage including all significant categories of speech acts.
- A flexible context.
- A semantic basis for meaning that emphasizes conventional meaning.
- A public perspective.

2.2.3.3 Dialogue Games

The modern study of formal dialogue systems for argumentation using dialogue games started with Charles Leonard Hamblin within the area of philosophical logic. Dialectic is the field of research concerned with the study of the dialectical contexts in which arguments are put forward [84]. Dialectical systems are normative models of dialogue which consists of [84]:

1. a set a moves e.g. challenge, assertion, question;
2. one commitment store for each conversant;
3. a set of dialogue rules regulating the moves; and
4. a set of commitment rules defining the effect of the moves on the commitment stores.

Research on interaction protocols based on dialectic argumentation is based mostly on speech act theory [160] and the philosophy of communicative action [82]. Informal dialogue studies in the area of philosophy [84, 103] have provided the core elements to design formal dialogue communication protocols e.g. [135]. Argumentation-based dialogues have been studied in various contexts such as within multi-agent systems [13, 95] for various purposes and also within AI and Law [18, 147].

Agent communications languages and protocols are formal languages, and their properties can be defined by semantic relationships to mathematical structures. However, because they are also intended as media for communication, each agent using a particular communications protocol will wish to ensure that all users share a common semantics or understanding of utterances [105]. Some examples of semantics used for agent communication protocols are:

- *Axiomatic semantics* [148] define each locution of a communications language in terms of the preconditions which must exist before the locution can be uttered, and the post-conditions which apply following its utterance. Axiomatic semantics can be public (known to all participants) or private (at least some of the pre- or post-conditions describe states or conditions which are internal to the participants).
- *Operational semantics* consider the locutions as instructions which operate successively on the states of some abstract machine. The semantics define the locutions in terms of the transitions they affect on the states of this machine.
- *Denotational semantics* assign each element of the language syntax to an abstract mathematical entity, its denotation. The semantics defines the locutions in terms of the transitions they effect on the states of this machine[9].
- In *game theoretic semantics* [88] each well-formed statement in the language is associated with a formal game between two players. A statement is considered to be true when a winning strategy exists for a player in the associated game. A winning strategy for a player is a rule giving that player moves for the game such that executing these moves guarantees the player can win the game, no matter what moves are made by the opposing player.

[9]Kripkean semantics [141] assigned to modal languages is an example of such a semantics.

Dialogue Game systems provide principles of coherent dialogue where the coherence is given in terms of a dialogue goal and the semantics of the utterances are defined at the dialogue level [38] . Dialogue games are defined as games between participants through a dialogue in the form of locutions where the participants have an objective and a finite set of moves to reach the objective [111]. In other words dialogue games are rule-governed interactions between two or more players where each player makes a "move" (utters a locution) complying to a set of rules.

As it has been pointed out, communication is not only the exchange of messages based on some rules and formats. Agents may need to converse because some inconsistency is present in the information available or simply to exchange information. Thus, we need to provide agents with mechanisms to handle inconsistency. Argumentation is an effective tool to deal with inconsistency because it involves agents putting forward arguments for and against propositions together with justification for the acceptability of these arguments [199]. This ability to question arguments gives argumentation-based communication languages a degree of verifiability that semantics relying on internal propositional attitudes lack. So, argumentation-based communication allows us to define a form of rationality in which agents only accept statements which they are unable to refute.

A dialogue game specification comprises the following elements [111]:

- Commencement Rules: Rules which define the circumstances under which the dialogue commences.
- Locutions: Rules which indicate what utterances are permitted. The dialogue game rules permit participants to utter propositions to which they assign differing degrees of commitment, for example: one may merely *propose* a proposition, a speech act which entails less commitment than would an assertion of the same proposition.
- Rules for Combination of Locutions: Rules which define the dialogical contexts under which particular locutions are permitted or not, or obligatory or not.
- Commitments: Rules which define the circumstances under which participants incur dialogical commitments by their utterances, and thus alter the contents of the participants associated commitment stores.
- Rules for Combination of Commitments: Rules which define how commitments are combined or manipulated when utterances incurring conflicting or complementary commitments are made. These rules become particularly important when multiple dialogues are involved, as when one dialogue is embedded within another; in such a case, the commitments incurred in the inner dialogue may take priority over those of the outer dialogue, or vice versa [66].
- Rules for Speaker Order: Rules which define the order in which speakers may make utterances.

- Termination Rules: Rules that define the circumstances under which the dialogue ends.

I present now dialogue types useful to understand the purpose and characteristics of dialogues.

2.2.3.4 Walton and Krabbe Dialogue Types

Agents need to negotiate to solve the problems posed by their interdependence on one another [134]. Negotiation provides a solution to these problems by giving the agents the means to resolve their conflicting objectives and correct inconsistencies in their knowledge of other agents' world view [134]. In the early papers on argumentation in multi-agent dialogues e.g. [134, 136], the authors described what the agents did as negotiation (the dialogues were about sharing some resources) before the typology of theorists Douglas Walton and Erik Krabbe was adopted to characterize agent dialogues.

Walton and Krabbe analyse the concept of commitment in dialogue [190] to provide conceptual tools for the theory of argumentation and define a set of dialogue types which has served as a basis for developers of MAS to design argumentation-based dialogues. The Walton and Krabbe typology has been useful to give a classification of multi-agent dialogues according to the purpose of what agents try to achieve individually and collectively. The typology helps to put the purpose and participants' aims in each type of dialogue in a specific context but recognized that an exchange between agents would typically include phases of different types of dialogue.

This classification is based upon three factors: (1) the information available to the participants, (2) the goal of the dialogue itself and (3) the individual goals of the participants. The list provides a sound basis to group communication scenarios between autonomous agents, though the authors do not claim this list to be exhaustive. Recently, in [189], Walton added to the initial list a new basic type "Discovery dialogue" as presented in [107]. This type of dialogues are normative models so they do not necessarily correspond to real instances of any of the original dialogue types. The seven primary dialogue types [189] are:

1. **Information-seeking Dialogues**: one participant seeks the answer to some question(s) from another participant, who is believed by the first to know the answer(s). This is the simplest kind of dialogue. A simple database search will be an example.

2. **Inquiry Dialogues**: participants collaborate to answer some question(s) whose answers are not known to any one participant e.g. [29].

3. **Persuasion Dialogues**: one party seeks to persuade another party to adopt a belief or point-of-view or to endorse a statement the participant does not currently hold. These dialogues begin with one party supporting a particular statement which the other party to the dialogue does not, and the first seeks to convince the

second to adopt the statement. Examples in the context of multi-agent systems can be found in [10, 24, 142].

4. **Negotiation Dialogues**: the participants bargain over the division of some scarce resource in a way acceptable to all, with each individual party aiming to maximize his or her share. Examples in the context of multi-agent systems can be found in [113, 132].

5. **Deliberation Dialogues**: participants collaborate in order to decide what action or course of action to take in some situation. Participants share a responsibility to decide the course of action, and either share a common set of intentions or a willingness to discuss rationally whether they share intentions. Examples in the context of multi-agent systems can be found in [95, 106, 171].

6. **Discovery Dialogues**: participants need to find an explanation of facts, the goal is to choose the best hypothesis where participants find and defend suitable hypotheses.

7. **Eristic Dialogues**: participants quarrel verbally with each aiming to win the exchange, as a substitute for physical fighting. There are a few if any examples of these in MAS.

Figure 2.7 presents a way to determine the dialogue type according to Walton and Krabbe in [190]. Given that the classification is not exhaustive, an agent dialogue could be of another nature or could be comprised of a combination of several types. In [13] for example, the authors present a protocol that allows agents to engage in dialogues regarding commands, which does not fit into any of the categories by Walton and Krabbe. In open systems, the beliefs of agents may not coincide, and so their interactions will require dialogues involving all of information seeking, mutual inquiry and persuasion interactions. Similarly, their intentions may also not coincide, and so their interactions will require dialogues involving persuasion, negotiation and/or deliberation [190]. In the approach presented in this thesis, agents have to choose the best plan and modify it if necessary, given a conflict in their beliefs and values. The intended dialogue then is one where agents are able to engage in a combination of a persuasion and a deliberation dialogue. More specifically:

Persuasion dialogues: In its simplest variant, a persuasion dialogue involves one participant seeking to convince another(s) to accept a statement following a conflict of points of view via a dialectical process [190]. A point of view with respect to a proposition can be in favour, against or undecided. The participant that proposes a claim has the aim to persuade the other participant(s) to adopt its point of view. A conflict is then resolved if all parties come to share the same point of view. Persuasion also allows agents to persuade one another that an action should be done. More specifically, *disputes* or *conflicts of contrary opinions* can be

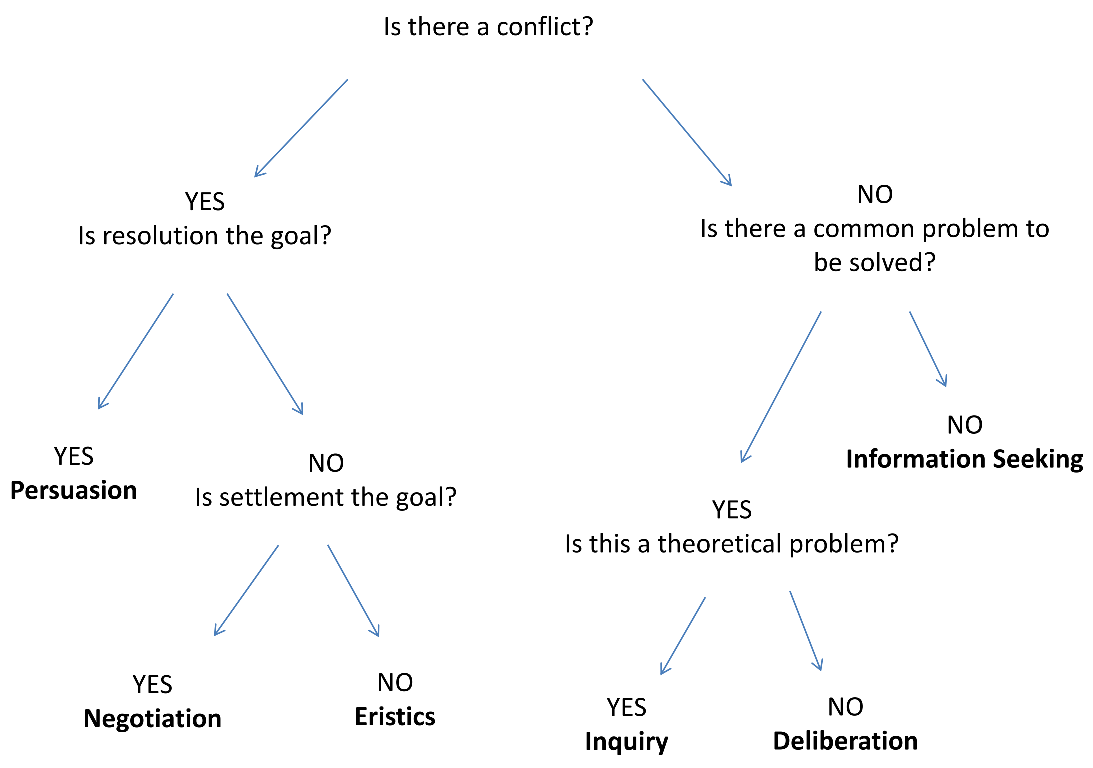


Figure 2.7: Determining the Type of Dialogue.
Image taken from [190] pp.81

seen as a subtype of persuasion dialogues where participants have contradictory points of view at the beginning of the dialogue rather than having no opinion on the matter. The following terminology used in a survey about persuasion dialogues by Prakken in [143] is useful to present the elements of persuasion dialogues:

- *claim* ϕ (assert, statement,...). The speaker asserts that ϕ is the case.
- *why* ϕ (challenge, deny, question). The speaker challenges that ϕ is the case and asks for reasons why it would be the case.
- *concede* ϕ(accept, admit,...). The speaker admits that ϕ is the case.
- *retract* ϕ (withdraw, no commitment, ...). The speaker declares that he is not committed any more to ϕ.
- *since* ϕ(argue, argument, ...). The speaker provides reasons why ϕ is the case.
- *question* ϕ. The speaker asks another participant's opinion on whether ϕ is the case.

Examples of the theory of persuasion dialogues are discussed in: Walton and Krabbe [190] (PPD "permissive"" and RDP for "rigorous" persuasion dialogues), Amgoud [6], Parsons and McBurney [131], Prakken [142] and Bentahar et.al. [24].

Deliberation dialogues: Deliberation dialogues occur when two or more participants seek to jointly agree on a course of action [106]. In a typical deliberation dialogue agents make proposals for action and move them in the dialogue. If the proposal is acceptable for the audience it is accepted and the dialogue moves to the next proposal. The course of action acceptable to an agent may be selected on the basis of considering preferences or goals. Deliberation dialogues are characterised by the absence of a fixed initial commitment by any participant and the chosen action is supposed to be acceptable to all concerned [106]. Nevertheless the participants have preferences and the actions they propose express individual positions about what is to be done, the discussion is conceptualized to be a mutual one directed at reaching a joint decision over a course of action. Indeed the line between deliberation and persuasion is very thin: the main difference is the perspective from which agents enter into the dialogue. Agents engaged in deliberations are characterized by having a *mutual* goal.

2.2.3.5 Argumentation-based Dialogue Protocols

Argumentation-based dialogues are used to formalize dialogues between autonomous agents based on theories of argument exchange. This section presents three argumentation-based protocols that are the basis of the Dialogue Game Protocol presented in Chapter 4. The protocols discussed are: the Fatio protocol in [110], the PARMA protocol in [10] and the Command Dialogue Protocol in [13].

- **The Fatio Protocol**

  In [110], McBurney and Parsons propose locutions to allow agents to give and receive reasons for statements with the *Fatio* interaction protocol. Essentially the protocol presents locutions necessary for argumentation that the *FIPA-ACL* lacks. The protocol allows participants to make assertions, request justifications for assertions, make challenges to assertions, provide justications (or arguments) for assertions, and retract prior assertions. The protocol classification of illocutions draws from speech act theory [161] and the philosophy of communicative action [82]. The locutions of the *Fatio* protocol are:

  - *assert*
  - *question*
  - *challenge*
  - *justify*
  - *retract*

  The Fatio protocol provides the means to undertake run-time assessments of claims made by agents in a dialogue. In this manner claims are contestable by participants and assessed for example for consistency, sincerity and so on. This process of assessment is not described in the protocol and further mechanisms (such as critical questions) could be used (in Chapters 3 and 4, I define a mechanism to do so).

- **The PARMA protocol**: In [10], Atkinson et. al. present the "Persuasive ARgument for Multiple Agents (PARMA) protocol". The protocol focuses on rational interactions between agents engaged in joint practical reasoning for persuasion over action that enables participants to propose, attack and defend an action or course of actions. Interactions are characterized by action proposals where one agent endorses a particular action and seeks to have another agent do the same. Agents put forward arguments for action with the use of the argumentation scheme for action proposals presented in Section 2.3.4. A position proposing an action may be attacked in a number of ways using list of rational attacks based on critical questions associated with this scheme. How a proponent of a proposal responds to an attack depends upon the nature of the attack which can be a factual disagreement, different preferences, differences of language representation or may seek clarification of a position.

- **Command Dialogue Protocol (CDP)**

  Another example of a protocol based on argumentation is the Command Dialogue Protocol (*CDP*) in [13]. The *CDP* is a representation of imperatives or commands for computational systems and a multi-agent dialogue protocol. To facilitate the dialogue the *CDP* is vested with a set of critical questions whose answers may defeat the initial argument (command). *CDP* uses the argument scheme for action proposals in [11] in the context of a command dialogue where commands are treated as presumptive arguments for action to be executed by a designated agent.

  The *CDP* allows an agent acting as the Commander to issue a command to another agent acting as Receiver, and allows Receiver to question, challenge, refuse or accept this command. Commands are instructions issued by one agent to one or more other agents to execute some action (or not), or to bring about some state. Not all commands are issued legitimately, and even those which are legitimate may require subsequent elaboration or explanation before they can be executed. Thus, it is possible for agents to engage in an argumentative interaction over a command. In contrast with proposals or promises, commands require a set of preconditions in a regulatory environment to be executed validly. A command represents a presumptive argument attacked by a set of critical questions whose answers may defeat the initial argument or command.

  Command dialogues are not explicitly mentioned in the Walton and Krabbe typology of human dialogues [190]. In a dialogue where a command has been issued, but not yet refused or accepted, the participants may enter into interactions which resemble those in the Walton and Krabbe typology e.g. persuasion, deliberation.

negotiation etc. However, not all command dialogues will have all such interactions, so that it is appropriate to consider command dialogues as a type of dialogue distinct from those in the Walton and Krabbe list.

Critical questions represent questions the Receiver could pose to the “Commander” either to question or to challenge the command such that more evidence will be needed to justify it. If questioned or challenged, the “Commander” can respond with additional information or arguments in support of the original command, and/or re-iterate it, modify it, or retract it. Questions about the appropriateness, suitability, feasibility and normative rightness can be posed to the “Commander”.

2.2.3.6 Desiderata for Agent Communication Protocols

In [112] McBurney et al. present a list of desirable features for communication protocols based on argumentation. Based on this list the authors assess various dialogue game protocols [7, 51, 109] and the *FIPA-ACL*. The proposed desiderata is as follows:

1. Stated Dialogue Purpose: A dialectical system must have a publicly-stated purpose(s) accepted by all the participants.
2. Diversity of Individual Purposes: A dialectical system must take into account the diversity of individual purposes and allow them to achieve their individual goals.
3. Inclusiveness: A dialectical system must be inclusive and not exclude any potential qualified agent.
4. Transparency: A dialectical system must be transparent exposing the rules and structure to all participants prior to the commencement of the dialogue.
5. Fairness: A dialectical system must be fair to all participants and treat them equally.
6. Clarity of Argumentation Theory: A dialectical system should conform to a stated theory of argument.
7. Separation from Syntax and Semantics: There must be a clear separation between syntax and semantics.
8. Rule-consistency: The rules of a dialectical system should be internally consistent.
9. Encouragement of Resolution: The dialectical system should facilitate normal termination of the dialogue and not preclude it by the rules or locutions.
10. Discouragement of Disruption: The dialectical system rules should discourage or preclude disruptive behaviour.
11. Enablement of Self-Transformation: The dialectical system should enable participants to change their preferences and drop previous commitments.

12. System Simplicity: The locutions and rules of a dialectical system should be as simple as possible.

13. Computational Simplicity: The locutions and rules of a dialectical system should be as simple as possible to minimize computational demands.

Probably the most important characteristic is that argumentation protocols should enable participants to change their preferences and drop previous commitments. For example, the weakness in the negotiation protocol in [7] is the absence of self-transformation capability. The dialogue protocol does not allow for preferences to be expressed in the dialogue; nor are degrees of belief or acceptability in propositions and arguments expressible.

In the protocol of Dignum et al. in [51] seeks to persuade opponents in the team to adopt a group belief or intention. This protocol embodies a theory of argumentation which is not necessarily appropriate for agent dialogues [112]. The theory assumes participants are engaged in a critical persuasion, where the rules and locutions are stricter than in a usual persuasion dialogue.

The *FIPA-ACL* is also analysed in [112]. In essence, *FIPA-ACL* does not allow self-transformation and hence gives no value to argumentation activities such as information-seeking, persuasion, inquiry or joint deliberation. As discussed earlier the rational-choice model of *FIPA-ACL* does not allow agents to engage in forms of dialogue that enable joint determination of plans of action.

Elements six (Clarity of Argumentation Theory) and eleven (Enablement of Self-Transformation) allow agents to express explicitly a particular view of joint decision making. Furthermore, new evidence may arise that can change the beliefs of an agent. I will use these criteria to assess the Planning Dialogue Game Protocol presented in Chapter 4.

2.3 Practical Reasoning and Argumentation

Practical reasoning is the use of reason to decide how to act (what to do) in a given situation. For an autonomous agent it is essential to choose its action by selecting the best possible option and justify its actions according to the environment. This section presents notions on how autonomous agents conduct practical reasoning using argumentation.

This section is structured as follows: Subsection 2.3.1 presents foundations of practical reasoning in philosophy. Subsection 2.3.2 presents definitions and types of argumentation schemes and critical questions.Subsection 2.3.3 presents how agents use argumentation schemes yo perform practical reasoning. Subsection 2.3.4 presents an argumentation scheme for action proposals (which is the basis of my argumentation scheme for plan proposals in Chapter 3). Finally, Subsection 2.3.5 presents action-state semantics for practical reasoning and Action-based Alternating Transition Systems, used to provide a semantic structure for the argumentation scheme for action proposals and its critical questions.

2.3.1 Practical Reasoning

Practical reason is the general human capacity for resolving, through a reflexive process, what one is to do [152]. A definition more related to autonomous agents is given in [33] by Bratman : "Practical reasoning is a matter of weighing conflicting considerations for and against competing options, where the relevant considerations are provided by what the agent desires/values/cares about and what the agent believes."

According to Walton[191], practical reasoning is a kind of goal-directed, knowledge-based reasoning that is directed to choosing a prudent course of action for an agent aware of its current circumstances.

There are three main views about the outcome or conclusion of practical reasoning, being: (1) an action, (2) a desire or an intention and (3) a belief about reasons for action [165].

In philosophy, practical reasoning has been discussed since the time of Aristotle using mainly the concept of practical syllogism ([161] presents a recent approach). A syllogism is a three-term argument which consists of a major premise, a minor premise, and a conclusion. For example:

Major premise: Friends ought to care for one another.
Minor premise: Gloria is a friend of mine about to commit a huge mistake.
Conclusion: I should talk to her and ask her to reconsider.

Further work in this field presents three points of criticism to the use of this abductive form of reasoning (taken from [11]):

C1 : There may be alternative ways to achieve the conclusion

C2 : Performing an action typically precludes other actions which may have better results

C3 : Performing an action may have undesirable consequences.

From these critique points we can say that people reason differently according to different choices of goals and actions. Practical reasoning is a subjective process that takes into account our desires and values as opposed to reasoning solely about our beliefs.

Searle [161], extended the practical syllogism adding a consideration for the “best” option and considering “all the possible things”, but this too presents problems as the consideration of “all things” is not generally possible and the concept of “the best option” is rather vague and subjective. Thus, Searle’s practical syllogism [161] is usually not enough to give agents the ability to reason accordingly in a given situation. For autonomous agents, in order to reason about what to do they have to deal with several constraints not captured by the practical syllogism such as the *best* option to execute, other possibly better alternatives and consequences not foreseen.

The Belief, Desire and Intention (BDI) architecture [151] is an approach to study agent oriented systems that represents the information (Beliefs), the motivations (Desires) and the plans or deliberation procedures (Intentions) of an agent. These “mental attitudes” determine the behaviour of agents in real-time performance scenarios where symbolic reasoning and a decision theoretic perspective is required [151]. The BDI agent architecture uses a standard way to justify an action in terms of beliefs, desires and intentions *e.g.* if an agent *desires* to achieve a goal G and *believes* action A realises goal G, therefore the agent has a candidate intention to perform A. The agent will then commit to choose one of the candidate intentions to commit to.

The practical reasoning process is called the “Deliberation Process” in [202] that consists of two main phases. The first is the *option generation* phase where the agent generates a possible set of actions to execute using its current beliefs and desires. The second is the *filtering* phase where the agent chooses the best option through, typically a utility function.

While this approach provides some solutions to the issues related to practical reasoning such as dealing with the best option through an utility function and dealing to some extent with the side effects of a plan through pre-approved plans, this is not sufficient [11]. With this approach the designer takes responsibility for determining the options available to agents, but because practical reasoning is open ended, unforeseen alternatives and consequences may occur at any time in a multi-agent environment.

Another approach to practical reasoning for autonomous agents is based on the instantiation of argumentation frameworks to generate desires and plans to achieve these desires in [149]. The worth of desires and the cost of resources are compared based on decision theoretic notions and integrated into three argumentation frameworks. In this thesis we consider the process of practical reasoning for autonomous agents as a reasoning process choosing an action to perform using *argumentation schemes and critical questions.*

2.3.2 Argumentation Schemes and Critical Questions

Artificial Intelligence has become increasingly interested in *Argumentation Schemes* due to their potential for making significant improvements in the reasoning capabilities of artificial agents and for the automation of agent interactions [23]. Argumentation schemes are stereotypical patterns of defeasible reasoning where arguments are presented as general inference rules from which, given a set of premises a conclusion can be presumptively drawn [186]. Many types of argumentation schemes have been studied (in [186] for example, there are listed more than thirty types of argumentation schemes) and some work has been done to formalize the logical structure and create computational models based on these schemes (examples can be found in [11, 77, 99, 130, 174]). In [77] for example, Carneades[10] uses computational models of various argumentation schemes to construct arguments using resource-bounded heuristic strategies to search the space of arguments. Some examples of argumentation schemes taken from [186] are:

- Argument from Expert testimony, example: e.g. *E is an expert in domain D, E asserts that A is known to be true, A is within D, therefore A may plausibly be taken to be true.*
- Argument from Evidence to Hypothesis, example: e.g. *If A is true (hypothesis), then B will be observed to be true (evidence), B has been observed to be true in a given instance, therefore A is true*
- Argument from Commitment, example: e.g. *a is committed to proposition A, therefore in this case a should support A.*
- Argument from Analogy, example: e.g. *Generally, case C_1 is similar to case C_2, A is true in case C_1 therefore A is true in C_2.*
- Argument from Popularity, example: e.g. *If large majority accept C as true, then there exists a (defeasible) presumption in favour of C. A large majority accepts C as true. therefore, there exists a presumption in favour of C.*
- Argument from Precedent, example: e.g. *The existing rule says that for all x, if x has property F then x has property G. But in this case C, a has property F but does not have property G. Therefore, the existing rule must be changed, qualified, or given up, or a new rule must be introduced to cover case C.*
- Argument from Example, example: *In this case, individual A has property F and property G, A is typical of things that have F and may or may not have G, therefore, generally if X has F, X has also G.*

[10]Carneades is an Open Source software library for building tools supporting a variety of argumentation tasks, including: argument construction from defeasible rules, argument evaluation, argument visualisation, and argument interchange.

Each of the schemes is associated with a set of characteristic critical questions. Critical questions and their responses are a way to examine argument acceptability when instantiating the related schemes. The real value of having associated critical questions with an argumentation scheme is that the questions enable an opponent to seek points of challenge in an argument or to locate premises in it that require justification. There are several ways to use Critical Questions in computational models for agent reasoning:

- An agent may use the questions as an internal mechanism to create arguments or enhance an argument proposal.
- An agent may use the questions as a way to generate arguments and attacks.
- In the context of a dialogue, questions could be used to attack an instantiated element of the argumentation scheme by asserting a negative answer to the question.
- Another way is to pose the questions to challenge assumptions so that the proponent has the dialectic obligation or burden of proof to provide more evidence than the presented.

The following subsections discuss the use of Argumentation Schemes and Critical Questions to generate attacks to arguments and in a dialogical context.

2.3.2.1 Critical Questions and Attacks

An instantiation of a critical question could be seen as a counterargument that seeks to refute the original argument posed by instantiating the argument scheme. Depending on the nature of the critical question, they can be used to critique several aspects of the argument. Most of the critical questions represent a valid way to challenge arguments that could identify sources of disagreement about a particular element of the argumentation scheme. A question can be seen as a weak form of attack on a particular element of the argument scheme given different beliefs about the world of the agent posing the question, since it questions its presence rather than asserts its absence.

Specifically, critical questions represent a way to evaluate arguments constructed using an argumentation scheme and provide pointers to ways in which the argumentation scheme can be shown to be inapplicable, thus suggesting a valid way to attack the argument, either defeating the argument on one of its premises or on its presumptive conclusion or showing the use of the scheme to be inappropriate in the context. These questions could focus for example on the premises and conclusions of the argument searching for evidence that may defeat the argument.

The classification of critical questions in [181] gives four different kinds of roles to the questions and provides a notion on how critical questions can be constructed:

- They can be used to question whether a premise holds.
- They can point to exceptional situations in which the scheme should not be used

- They can set conditions for the proper use of the scheme.
- They can point to other arguments that might be used to rebut the claim.

In [77], critical questions are modelled as additional premises of an argument classified as either assumptions or exceptions. The distinction between assumptions and exceptions allows some answers to be assumed for critical questions which have yet to be asked. Critical questions then can be constructed from these premises and this classification can indicate how to shift the burden of proof in the dialogue generated from this argument proposal. The *burden of proof* is the obligation that bears on a speaker to prove or give evidence to something stated in a dialogue. A question coming from an "exception premise" need to be backed up by an argument that the exception holds. A question that comes from an "assumption premise" requires the proponent to establish the truth of the assumption. Questions in this category could refer to the validity or legitimacy of variables. For example an argument scheme based on an expert opinion assumes that the referred expert is a credible source, and so the proponent must be able to justify this if requested to do so.

2.3.2.2 Argumentation Schemes, Critical Questions and Dialogues

In a dialogical context, critical questions allow the creation of rich and sustained dialogue sequences between a "proponent" and a "respondent". In [188], Walton explains: *"...arguments need to be examined within the context of an on-going investigation in dialogue in which questions are being asked and answered"*.

Argumentation schemes have a dialogical aspect because each is associated with its own set of characteristic critical questions, which indicates the conditions under which it can be properly used and provides pointers to challenge the arguments created using the scheme [77]. A benefit of having critical questions in a dialogue is that the questions enable participants to identify points of challenge in a debate or locate premises in an instantiation of the argument scheme that can be recognised as questionable. Computationally, critical questions have been used to create dialogue protocols for agents where the participants put forward arguments instantiating the argumentation scheme and opponents to the argument challenge it through objections based on critical questions e.g. [10, 39, 86, 157, 172, 173].

In dialogue protocols for autonomous agents there should be rules embedded in the protocol on how the burden of proof should be handled between participants. When evidence that complies with the claim made is provided, we say that the burden of proof has been satisfied. In [145], Prakken et al. consider two levels of burden of proof in a dialogue: the *global level*, where the burden of proof pertains to a participant's ultimate goal in the dialogue and is fixed and the *local level*, where the burden of proof may change during the dialogue and the rule to change depends entirely on the norms established in the social context. How the burden of proof shifts on a dialogue depends on domain specific issues and the domain context in which the dialogue takes place [145].

For some critical questions, merely asking the question is enough to shift the burden of proof back to the party who put forward the argument to answer the question. For some other questions this is not enough and that the question should be combined with some evidence of the alleged bias. The burden of proof thus is an important factor to consider when creating computational models of dialogue for autonomous systems. Furthermore, the burden of proof then can become the subject of debate during a dialogue [145].

Dialectically a question should be treated as a locution where an answer is expected. When asking a question in a dialogue an agent may be looking to challenge an element for which it has a proof that it is false (and accept the burden of proving this). Or, if the agent has no proof, it may pose the question and expect the burden of proof is on the opponent [145]. If the question itself poses an attack to another argument it should be treated as a challenge[11]. Critical questions then, can be conceptualised as attacks in a typical argumentative dialogue if the speaker presents a proof related to the element questioned.

In Chapter 3, an analysis of questions that match an argumentation scheme for plan proposals is presented, and in Chapter 5 we provide a deeper analysis on the nature of questions to create a strategy to select questions in a dialogue.

2.3.3 Practical Reasoning using Argumentation Schemes

There is an important difference when arguing about beliefs and what should be done. In essence how to act may differ between agents since agents have different aspirations and values. Some related work on practical reasoning for agents relies on the use of rules to express the consequences of actions, and the conditions under which a goal will be desired (e.g. [22] and [149]). The approach in this thesis is based on the work of Atkinson and Bench-Capon in [8]. In [186], Walton provides a definition of practical reasoning that is close to the concept of practical reasoning for autonomous agents:

"Practical reasoning is a goal directed sequence of linked practical inferences that seeks out a prudent line of conduct for an agent in a set of particular circumstances known by the agent."

In [191], two basic types of practical inferences related to practical reasoning are discussed:

- the necessary condition scheme i.e. G is a goal for agent ag, doing A is necessary to carry out G, therefore, agent ag ought to do A A is an action and G is a goal .
- sufficient condition scheme i.e.v G is a goal for A, doing A is sufficient to carry out G, therefore, ag ought to do A.

[11]Under this principle is based the Dialogue Game Protocol presented in Chapter 4.

With this scheme the presumptive reason for performing the action can be challenged and withdrawn with the use of four critical questions:

1. Are there alternative ways of realizing goal G?
2. Is it possible to do action A?
3. Does agent ag have goals other than G which should be taken into account?
4. Are there other consequences of doing action A which should be taken into account?

The notion of goal in this schemes is rather ambiguous, Is the goal G the premise? Is the goal G the result of the action? Is the goal the consequence of the action? Is the goal the reasons why these consequences are desired?

In [10], these differences are discussed and Walton's notion separated into three distinct elements: states, goals and values, where *states* are sets of propositions about the world, *goals* are propositional formulae composed from this set of propositions and *values* are functions on goals conceptualised as the reason for which an agent wish to achieve a goal.

Practical reasoning can be conceptualised as a three stage process (furthermore, this is a way of classifying CQs) as follows [8]:

- Problem formulation: deciding what are the propositions and values relevant to the particular situation.
- Epistemic reasoning: determining the initial state in the structure formed at the previous stage.
- Choice of action: developing the appropriate arguments and counter arguments, in terms of applications of the argument scheme and critical questions, and determining the status of the arguments with respect to other arguments and the value orderings.

These stages may be carried out sequentially, or they may iterate if the critical questioning leads to a reformulation of the problem. This thesis adapts this three stage process in the context of a dialogue about plans. The choice of action becomes the process of agreeing and possibly modifying a plan with the use of the argument scheme and critical questions. The status of these arguments can be determined through the dialogue using a *VAF* 2.2.2.2. The next subsection presents an extension of Walton's scheme taking into account these elements.

2.3.4 Argumentation Schemes for Action Proposals

In [11], Atkinson et al. present an extension to Walton's scheme [191] considering states, goals and values. The argumentation scheme for action proposal *AS1* in [11] is presented in Table 2.2.

The authors of [11] assume the existence of:

TABLE 2.2: AS1: An Argumentation Scheme for Action Proposals.

In the current circumstances R,
we should perform action A,
to achieve new circumstances S,
which will realize goal G,
which will promote value v.

- A finite set of distinct actions (Actions), with elements, A, B, C, etc.
- A finite set of propositions (Props), with elements, p, q, r, etc.
- A finite set of states (States), with elements, R, S, T, etc. Each element of States is an assignment of truth values T, F to every element of Props.
- A finite set of propositional formulae called goals (Goals), with elements G, H, etc.
- A finite set of values (Values), with elements v, w, etc.
- A function value mapping each element of Goals to a pair $< v, sign >$, where v $\in$ Values and sign $\in \{+, =, -\}$.
- A ternary relation *apply* on Actions $\times$ States $\times$ States, with $apply(A, R, S)$ to be read as: "Performing action A in state R results in state S".

Walton's original critical questions are also extended considering the new elements to obtain sixteen questions in total which are the basis of the theory of persuasion over action in [11]. The critical questions are presented in the following list:

CQ1: Are the circumstances true?

CQ2: Assuming the circumstances, does the action A will bring about consequences S?

CQ3: Assuming the circumstances and that the action A will bring about consequences S will the action bring about the desired goal?

CQ4: Does the goal realise the value stated?

CQ5: Are there alternative ways of realising the same consequences?

CQ6: Are there alternative ways of realising the same goal?

CQ7: Are there alternative ways of promoting the same value?

CQ8: Does doing the action have a side effect which demotes the value?

CQ9: Does doing the action have a side effect which demotes some other value?

CQ10: Does doing the action promote some other value?

CQ11: Does doing the action preclude some other action which would promote some other value?

CQ12: Are the circumstances as described possible?

CQ13: Is the action possible?

CQ14: Are the consequences as described possible?

CQ15: Can the desired goal be realised?

CQ16: Is the value indeed a legitimate value?

2.3.5 Action State Semantics for Practical Reasoning

In [8], a semantics for practical reasoning is presented based on a formalisation that handles the temporal aspects of practical reasoning, and enables the justification of actions. The formalism presents a practical reasoning approach using states and actions with regard to temporal considerations. The formalism greater expressiveness allows the modelling problems where there are temporal windows of opportunity where coordination is required, and where the likelihood of an event varies with time.

An approach to model practical reasoning through argumentation temporal aspects and the intrinsic worth of actions is presented in [8]. Time is important because actions change the state of the world and when agents argue about joint-actions they consider future states as well as past and present ones. States can be reached by several actions through actions justified by the values they promote/demote.

2.3.5.1 Action-based Alternating Transition Systems

To define formally the argumentation scheme for action proposals and the associated critical questions, Atkinson and Bench-Capon used a structure based on Alternating Transition Systems[12] named Action Based Alternating Transition Systems (AATS) [178]. Such a structure is needed to reason about actions and their effects in terms of transitions from one state to another. In this way the effects of actions can be made dependent on the action of other agents, and other events in the environment. In [178], these structures are used to describe coordination in multi-agent systems using social laws (introduced through the work of Moses and Tennenholtz [125]).

An *AATS* models joint-actions that may be performed by a set *Ag* of agents in a state and the effects of these actions. The systems modelled with an AATS may be

[12]Alternating Transition Systems (ATS) were originally developed to underpin the Alternating-time Temporal Logic of [3].

in any of a finite set Q of possible states, with $q = 0 \in Q$ as the initial state. Each agent $i \in Ag$ is associated with a set Ac_i of possible actions unique to agents. A joint action j_C (coalition C) for set of agents, is a tuple $\{\alpha_1, ..., \alpha_k\}$ where for each α_j (where $1 \leq j \leq k$) there is some $i \in C$ such that $\alpha_j \in Ac_i$. Given an element j of J_C and an agent $i \in C$, is action in j is denoted by j_i .

Specifically, an *AATS* model defines semantic structures useful to represent joint-actions for multiple agents, their preconditions and the states that will result from the transition. An *AATS* is an (n+7)-tuple of the form:

$$S = \langle Q, q_0, A_g, Ac_1, ..., Ac_n, \rho, \tau, \Phi, \pi \rangle$$

where:

- Q is a finite non-empty set of **states**;
- $q_0 \in Q$ is the **initial state**;
- $A_g = \{1, ..., n\}$ is a finite non-empty set of **agents**;
- Ac_i is a finite, non-empty set of **actions**, for each $i \in Ag$, where $Ac_i \cap Ac_j = \emptyset$ for all $i \neq j \in Ag$;
- $\rho : Ac_{Ag} \rightarrow 2^Q$ is an action **precondition function**, which for each action $\alpha \in Ac_{Ag}$ defines the set of states $\rho(\alpha)$ from which α may be executed;
- $\tau : Q \times J_{Ag} \rightarrow Q$ is a **partial system transition function**, which defines the state $\tau(q, j)$ that would result by the performance of j from state q, note that, as this function is partial, not all joint actions are possible in all states (cf. the precondition function above);
- Φ is a finite, non-empty set of atomic **propositions**; and
- $\pi : Q \rightarrow 2^\Phi$ is an **interpretation function**, which gives the set of primitive propositions satisfied in each state: if $p \in \pi(q)$, then this means that the propositional variable p is satisfied (equivalently, true) in state q.

In [9], Atkinson and Bench-Capon extended this *AATS* model to enable the representation of a theory of practical reasoning related to arguments about action through which *values* were added to the system. The use of the term *value* follows [11] where values are qualitative social interests of agents. The extensions to the *AATS* model are:

- Av_i, is a finite, non-empty set of **values** $Av_i \subseteq V$, for each $i \in Ag$.
- δ: $Q \times Q \times Av_{Ag} \rightarrow \{+, -, =\}$ is a **valuation function** which defines the status (promoted (+), demoted (-) or neutral (=)) of a value $v_u \in Av_{Ag}$ ascribed by the agent to the transition between two states: $\delta(q_x, q_y, v_u)$ labels the transition between q_x and q_y with one of $\{+, -, =\}$ with respect to the value $v_u \in Av_{Ag}$.

2.3.5.2 Formalising Argumentation Schemes and Critical Questions

In [8, 9], the AATS is used to reformulate the problem establishing the relevant propositions and values stated in the argumentation scheme for action proposals $AS1$. The argumentation scheme AS1 in terms of the extended AATS presented in table 2.3. The formalism can be used to "produce arguments" for action to be performed on its own merits and deal with temporal aspects. The critical questions were also formalised in these terms as follows [8]:

- CQ1: Are the believed circumstances true? [$q_0 \neq q_x$ and $q_0 \notin \rho(\alpha_i)$]
- CQ2: Assuming the circumstances, does the action have the stated consequences? [$\tau(q_x, j_n)$ is not q_y]
- CQ3: Assuming the circumstances and that the action has the stated consequences, will the action bring about the desired goal? [$p_a \notin \pi(q_y)$]
- CQ4: Does the goal realise the value stated? [$\delta(q_x, q_y, v_u)$ is not $+$]
- CQ05: Are there alternative ways of realising the same consequences? [Agent $i \in Ag$ can participate in joint action $j_m \in J_{Ag}$, where $j_n \neq j_m$, such that $\tau(q_x, j_m)$ is q_y]
- CQ06: Are there alternative ways of realising the same goal? [Agent $i \in Ag$ can participate in joint action $j_m \in J_{Ag}$, where $j_n \neq j_m$, such that $\tau(q_x, j_m)$ is q_y such that $p_a \in \pi(q_y)$ and $p_a \notin \pi(q_x)$ or $p_a \notin \pi(q_y)$ and $p_a \in \pi(q_x)$]
- CQ07: Are there alternative ways of promoting the same value? [Agent $i \in Ag$ can participate in joint action $j_m \in J_{Ag}$, where $j_n \neq j_m$, such that $\tau(q_x, j_m)$ is q_z, such that $\delta(q_x, q_z, v_u)$ is $+$]
- CQ08: Does doing the action have a side effect which demotes the value? [In the initial state $q_x \in Q$, if agent $i \in Ag$ participates in joint action $j_n \in J_{Ag}$, then $\tau(q_x, j_n)$ is q_y, such that $p_b \in \pi(q_y)$, where $p_a \neq p_b$, such that $\delta(q_x, q_y, v_u)$is $-$]
- CQ09: Does doing the action have a side effect which demotes some other value? [In the initial state $q_x \in Q$, if agent $i \in Ag$ participates in joint action $j_n \in J_{Ag}$, then $\tau(q_x, j_n)$ is q_y, such that $\delta(q_x, q_y, v_w)$ is $-$, where $v_u \neq v_w$]
- CQ10: Does doing the action promote some other value? [In the initial state $q_x \in Q$, if agent $i \in Ag$ participates in joint action $j_n \in J_{Ag}$, then $\tau(q_x, j_n)$ is q_y, such that $\delta(q_x, q_y, v_w)$ is $+$, where $v_u \neq v_w$]
- CQ11: Does doing the action preclude some other action which would promote some other value? [In the initial state $q_x \in Q$, if agent $i \in Ag$ participates in joint action $j_n \in J_{Ag}$, then $\tau(q_x, j_n)$ is q_y and $\delta(q_x, q_y, v_u)$ is $+$ there is some other joint action $j_m \in J_{Ag}$, where $j_n \neq j_m$, such that $\tau(q_x, j_m)$ is q_z, such that $\delta(q_x, q_z, v_w)$ is $+$, where $v_u \neq v_w$]

- CQ12: Are the circumstances as described possible? $[q_x \notin Q]$
- CQ13: Is the action possible? $[j_n \notin J_{AG}]$
- CQ14: Are the consequences as described possible? $[\tau(q_x, j_n) \notin Q]$
- CQ15: Can the desired goal be realised? $[p_a \notin \pi(q)$ for any $q \in Q]$
- CQ16: Is the value indeed a legitimate value? $[v_u \notin V]$

Table 2.3: AS2: Formal AATS representation of the $AS1$ Argumentation Scheme.

In the initial state $q_0 = q_x \in Q$,
Agent $i \in Ag$ should participate in joint action $j_n \in J_{Ag}$
where $j_n^i = \alpha_i$
such that $\tau(q_x, j_n)$ is q_y,
such that $p_a \in \pi(q_y)$ and $p_a \notin\in \pi(q_x)$,
or $p_a \notin \pi(q_y)$ and $p_a \in \pi(q_x)$
such that for some $v_u \in Av_i$, $\delta(q_x, q_y, v_u)$ is $+$.

Figure 2.8 presents a simple transition system in AATS terms. The transitions are labelled with the values promoted and/or demoted by the execution of the actions. Each state corresponds to a set of possible worlds. A goal is represented as a formula the holds in a state. If the initial state is q_0 and the goal holds in state q_2 there are three ways to achieve the goal in this transition system: executing actions j_1 followed by action j_2 to promote values v_1 and v_3; executing action j_3 promoting value v_1 and executing action j_4 promoting value v_2. Using the argumentation scheme AS1 we can instantiate it to present an argument for action j_3 from state q_0 based on the promotion of v_1 to achieve state q_0 and achieve the goal. If v_1 is not a desired value then an alternative choice is to execute action j_4 and promote value v_2 to achieve the goal.

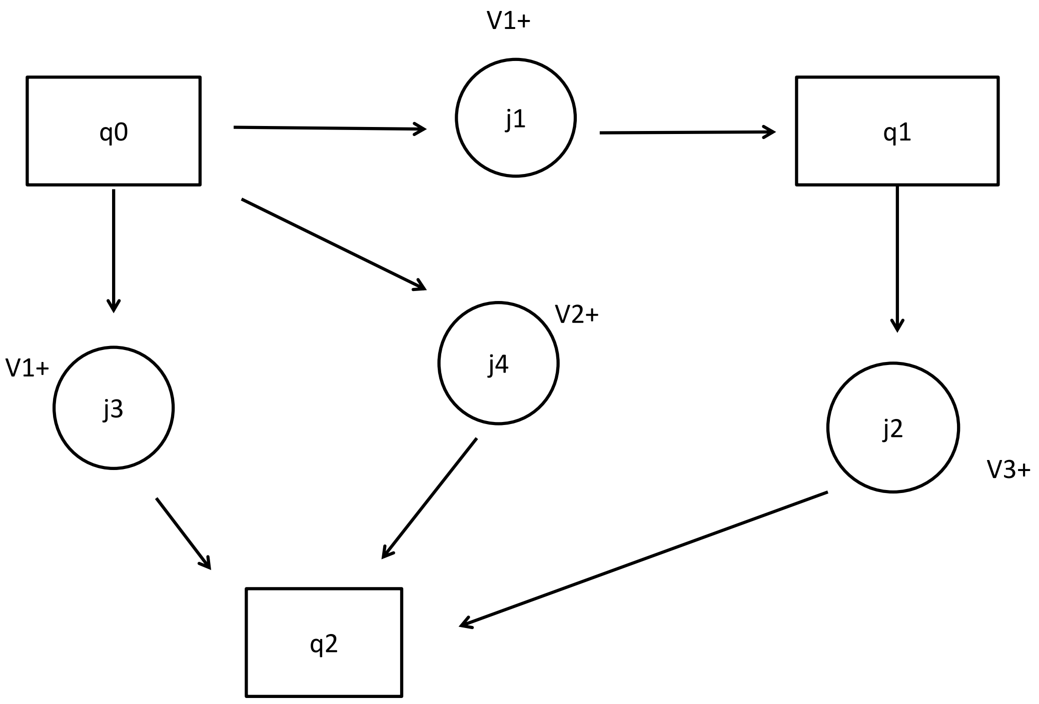


FIGURE 2.8: AATS Action State Semantics. Boxes represent states q_n and circles represent joint-actions j_n labelled by the promotion of some value v_n.

2.4 Dialogues and Planning

Cooperation is defined as a problem where a group of agents choose to work together to achieve a common goal [201]. Coordination on the other hand is defined as the process by which an agent reasons about its local actions and the (anticipated) actions of other agents to ensure the community acts in a coherent manner [91]. The problem of cooperation and coordination of multiple autonomous agents attempting to reach a common goal through the creation of a plan and consideration of the communication involved has been tackled in several ways in the past twenty years.

The environment of a MAS is the natural place for understanding and designing agent coordination [156]. For example blackboards [126, 127] are coordination artifacts that allow MAS coordination. Cooperation and coordination approaches cover using commitments and conventions as the basis for coordination e.g. [91], achieving coordination through the combination of combining joint planning and learning e.g. [194], achieving coordination through argumentation dialogues e.g. [132] etc.

The *multi-agent planning problem* has both a planning and a coordination component. Plans are not only used to guide action but also to control reasoning and enable agent interaction [139]. In this thesis plans are viewed as entities that guide the action of agents and enable agent cooperation.

This section presents core definitions on Artificial Intelligence Planning as an introduction on how my approach overcomes some multi-agent planning limitations using argumentation-based dialogues.

The rest of the section is organized as follows: Subsection 2.4.1 presents a brief summary on planning in AI and MAS to contextualize dialogue about plans. Subsection 2.4.2 presents a survey on cooperative approaches to planning and some communication characteristics. Subsection 2.4.3 focuses on argumentative dialogues approaches to planning related tasks.

2.4.1 Planning in Artificial Intelligence and *MAS*

Planning in Artificial Intelligence is concerned with the automatic synthesis of action strategies from a description of actions, sensors and goals [73]. The classical planning problem can be defined as follows [73]: Given

1. a description of the initial state of the world (in a formal language, usually propositional logic),
2. a description of the goal (i.e., a set of goal states) and
3. a description of the possible (atomic) actions that can be performed, modelled as state transformation functions,

determine a sequence of actions (a plan) that transforms the initial configuration of the world into one of the goal states. The formal language STRIPS[13] [68] is common to most classical planning frameworks.

Temporal planning is a branch of Artificial Intelligence Planning that deals with planning in situations where actions have non-zero duration, have an explicit representation of time and may overlap in time. Actions therefore need an explicit representation of time since they occur over a time span , therefore some implications of this are that the conditions of the action not only at beginning, the effects can be during or even after the action, actions may need to maintain partial states, events expected to occur in future time periods, goals must be achieved within time bound, etc. This notion naturally divides the space of temporal languages into those that can require concurrency (temporally expressive) and those that cannot (temporally simple) [71]. The problem of *Scheduling* related to planning is the problem of assigning the actions that need to be performed in the plan a timetable or a start time based on th duration and order of the actions.

It is important to differentiate between the following planning tasks (taken from [140] and [73]) when we refer to multi-agent planning (these tasks are not mutually exclusive):

- Plan generation: refers to the development of problem solvers either domain dependent or not.

[13]Note that STRIPS could also refer to an automated planner developed at Stanford Research Institute by the same authors in [68].

- Conflict identification: To create a plan the agents must have the ability to identify relevant conflicts given agents' views of the world may be different.
- Conflict resolution: The mechanism to resolve the conflicts identified.
- Plan sharing: refers to mechanism to share plans within a protocol.
- Environment monitoring: In dynamic contexts many events can occur that either indicate a problem with existing ordered actions or suggest a new path (possibly better) to take.
- Alternative assessment: It is the process of assess a plan related to its costs. Once determined a plan is valid, the assessment of the costs and benefits related to it.
- Coordination with other agents: In multi-agents contexts a wide range of issues arise involving the coordination of actions and plans.
- Plan modification: the process of modify plans that are partially correct. this mechanism can involve re-planning from the beginning or identifying points to back-track.
- Plan Monitoring: the mechanism to monitor the execution of the plan and modify it if necessary.

Multi-agent planning (MAP) involves coordinating the actions of multiple agents towards a common goal. This problem can take several variants depending where the planning is actually done e.g. one agent coordinating the plans or planning of others or agents refining their own plans while negotiating over tasks or resources [63].

In [193], the multi-agent planning problem is defined as follows :

"*The multi-agent planning problem is the following problem: Given a description of the initial state, a set of global goals, a set of (at least two) agents, and for each agent a set of its capabilities and its private goals, find a plan for each agent that achieves its private goals, such that these plans together are coordinated and the global goals are met as well.*"

The multi-agent case is not covered by the general planners because because multiple agents may have their own goals and preferences and it is often undesirable to create the plan for all agents centrally. These agents may wish to create their own plans independently or refuse to make all information necessary available to someone else. Furthermore in some cases dependencies between the actions of the agents make independent planning impossible, in other words, if the agents do not share heir action capabilities then they might come into conflict when they try to execute their plans. The complexity of distributed systems restricts the application of single-agent planning strategies to distributed problems usually because a local agent view is not sufficient. According to [166], effective *teamwork* in multi-agent situations requires:

- the establishment of **joint intentions**
- the determination of which **goals** to achieve
- the creation of a **joint-plan**
- the **sharing of knowledge** about the environment and
- the ability to **monitor the joint-plan execution**

Under this definition, the **sharing of knowledge** to determine intentions and plans is essential. Without communication the effective achievement of complex multi-agent goals and execution of actions is infeasible [195].

2.4.2 Cooperative Dialogues and Planning

Communication and *Reasoning* are inevitably present in advanced forms of cooperative teamwork because the problems need to be solved collectively and the agents solving them are not specifically designed to work together [58]. In this section I describe the use of cooperative dialogues in relation to planning related tasks. In multi-agent systems (MAS), especially in BDI (Belief, Desire, Intention) systems, teamwork is crucial [61]. Formal models of plans [41, 114] define a plan as a sequence of ordered actions. If the creation and sharing of plans were simply an exchange of a sequence of actions there would be no need to create a methodology to plan sharing and cooperation.

Agents exchange several "planning elements" when creating a plan and the coordination of these elements goes beyond establishing a sequence of actions. For example, agents need to agree on the overall goal, on sub-goals, on important and critical actions in the plan, on potential problems based on the beliefs of other agents etc. Several studies exploring how humans create a joint-plan and the way they communicate in real life to do so, have identified these elements, *e.g.,* [67, 80] and more recently in [169]. Similar conclusions and additions to planning considerations have been made in research that involves for example, robot planning [85] and plan modification and generalization [93].

The TRAINS project in [2], for example, is a case study in building a conversational planning agent. The participants involved in a planning dialogue process need to communicate to make sure that the plan is created and executed correctly. Based on natural language human-machine interaction, the system generates plans according to the feedback received form the interaction. The TRAINS system mixes natural language understanding and reasoning about time, actions and events to generate mixed initiative plans. Furthermore, it analyses how humans collaborate to form plans and the insights gained from this work will contribute to setting a basis for the argumentation theory in my research.

In [195], Werner develops a formal theory of high-level linguistic communication that serves as a foundation for understanding cooperative action in groups of autonomous agents. Werner relates the communication process to the intentions in an agent's mental model to generate the plans. The work also starts defining information with a formal abstract theory of knowledge representation.

In [167], Sycara presents a model of persuasive argumentation that has been implemented as part of a multi-agent system that operates in the domain of labour negotiations. Persuasive argumentation is used as a general mechanism for planning how to influence agents' intentions in order to increase cooperation. In [167], an agent's model is represented in terms of the agent's beliefs and preferences. The construction of arguments is performed using integration of case-based reasoning, graph search and approximate estimation of agents' utilities.

2.4.3 Argumentation-based Dialogues and Planning

The approach in this thesis is influenced by work on argumentation for practical reasoning [9], argumentative dialogue protocols [11, 13] and dialogues about plans [16, 128, 170].

In [13], for example, the dialogue focuses on a command exchange between participants where most of the critical questions inquire about a future state. This dialectical questioning about future states of the world lead me to fill the gap of questions and dialogues prior to a command scenario to a practical reasoning scenario where agents agree on a plan to execute. But questions about the plan itself are missing. This is the gap I am trying to cover.

In [58], Dunin-Keplicz and Verbrugge define *planning steps* in relation to dialogue. The first step is the discussion of proposals as a subtype of persuasion dialogue. The outcome of this dialogue should be a set of sub-goals and a common belief about these goals. The next step required is an *inquiry* or *deliberation* dialogue that matches actions with subtasks.

Persuasion and *information seeking* dialogues are then applicable during the allocation of these tasks. Even negotiation dialogues may be required because agents may have a conflict of interests during action allocation. Finally, a collectively trusted team member may conclude action allocation. Figure 2.9 shows possible dialogue embedding during planning.

As Dunin-Keplicz and Verbrugge suggest, planning tasks require several dialogue interactions (possibly embedded) for different steps in the planning process. The next chapters in this thesis will justify the use of a persuasion and deliberation dialogue when agents try to agree on previously created plans and possibly modify them.

Regarding dialogues and plans, Tang, Norman and Parsons in [170] establish a model for individual and joint actions suitable for describing the behaviour of a multi-agent team, including communication actions. The work of Tang et al. has focused on setting a basis for implementing multi-agent planning dialogues based on argumentation that takes into account the communication needs for the plan to be executed successfully. The model uses policies to generate plans and the communication needs are embedded in the policy algorithm generation. The approach in Tang et al. embeds the communication policy in the planning algorithm.

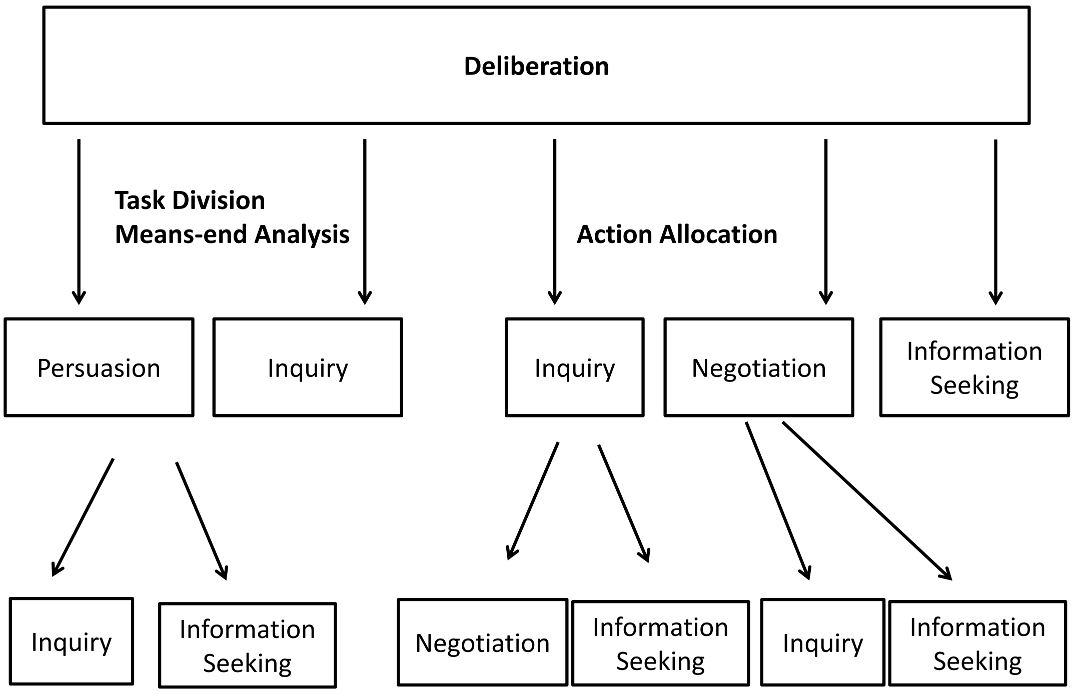


Figure 2.9: Possible Dialogue Embeddings during Planning Tasks. Image taken from [58] pp.126.

In [16], Belesiotis et al. develop an argumentation mechanism for reconciling conflicts between agents over plan proposals. The authors extend a protocol where argument-moves enable discussion about planning steps in iterated dispute dialogues as presented in [61]. The authors then introduce a logic for arguments about plans based on the situation calculus [114]. This work is focused on protocols to enable agents to discuss plans, resolve conflicts and reach agreements. Conflicts are caused by inconsistencies between beliefs regarding the state of the world or the specification of the planning operators. Argument moves enable discussion about planning steps and are integrated into a protocol for belief argumentation. The argumentation protocol allows the agents to discuss plan proposals and to identify reasons for potential disagreements that originate in differences regarding beliefs. This approach concerns resolving conflicts about beliefs, my approach is oriented to resolving conflicts about actions.

Another related approach is presented by Onaindía et al. in [128] where the authors present the problem of solving cooperative distributed planning tasks through an argumentation-based model. In this approach agents accomplish distributed planning tasks through an argumentation based model that allows agents to exchange partial solutions, express opinions on the adequacy of agents' solutions and adapt their own proposals for the benefit of the overall task. The argumentation based model is designed in terms of argumentation schemes and critical questions whose interpretation is given through the semantic structure of a partial order planning paradigm. The model allows agents to exchange partial solutions, express opinions on the adequacy of candidate solutions and adapt their own proposals for the benefit of the overall task. The approach assumes a lack of uncertainty and deterministic planning actions, thus, focuses only on questions concerned with the choice of actions. The argumentation scheme, based on the scheme for action proposal from [10] is of the form: *In the current circumstances and considering the current base plan Π_i, agent ag_i should perform the refinement step Π', which will result in a new partial plan Π_j, which will realise some sub-goals G, which will promote some values V.*

My approach is influenced by practical reasoning using argumentation schemes and dialogue game protocols. I believe plans are entities that need to be discussed and agreed at a detailed level when arguing about them. The use of argumentative dialogues presents advantages to generate, share and manage plans for autonomous agents that I will discuss in the next chapters. In the approach presented in this thesis agents reason over previously created plans and engage in a dialogue to justify the actions and possibly modify the plan. The main goal is to develop methodologies to allow agents to engage in dialogue for collaborative planning based on the practical reasoning capabilities of the agents.

2.5 Chapter Summary

I presented in this literature review core concepts on which the research on this thesis is based. The demands of MAS require planning-agents to communicate and manage their plan resources effectively. Furthermore, the complexity of MAS limits the use of single-agent planning strategies in distributed problems because the local beliefs of an agent are often not sufficient to generate a satisfactory plan.

Argumentation in Artificial Intelligence is a formal discipline that tries to implement models of argument used by humans through computational models to enable agents to cooperate and coordinate efficiently in order to reach their goals. Argumentation provides a powerful framework for interacting agents taking decisions, assessing the validity of information, or otherwise resolving differences of opinion.

In particular, Argumentation has been used successfully in AI to build computer systems able to exchange arguments to establish *dialogues* as a medium of negotiation, persuasion, deliberation, etc. where agents try to persuade other parties to accept offers through argumentation. In particular, research in argumentation and *automated dialogues* is useful to define how self-interested autonomous agents can interact with one another. The categorization of dialogues given by Walton and Krabbe has inspired researchers in MAS to design and implement models of communications based on argumentation theories. The correctness and implementation of teamwork dialogues is a key issue for this approach. This is the area in which the research on this thesis focuses.

The practical reasoning process in autonomous agents consist of agents generating a possible set of actions to execute using its current beliefs and desires and choosing the best option. One way to model practical reasoning in autonomous agents is using argumentation schemes due to their potential for making improvements in the reasoning capabilities of agents and the automation of agent interactions using critical questions. Planning tasks in MAS require communication, and the exchange of beliefs desires and intentions through interactions in different steps in the planning process. The following chapters present concepts on how agents can engage in interactions to propose, evaluate and refine plans in a multi-agent scenario.

Chapter 3

A Framework for Plan Proposals

This chapter presents an ***A****rgumentation* ***S****cheme for* ***P****lan proposals* (ASP) to propose multi-agent temporal plans based on the promotion or demotion of values. The multi-agent plans considered are conceptualized as sequences of scheduled actions for each agent involved in the achievement of the goal. The argumentation scheme presented builds on an argumentation scheme that allows agents to create proposals for action based on the promotion or demotion of values in [11]. The objective is that autonomous agents can use the ASP scheme and its associated critical questions to evaluate multi-agent plans.

Contrasting with the approach in [11], the type of actions I consider are *durative*, that is, actions have a start time and a duration. Durative actions were originally defined in the context of temporal planning. Together with the argumentation scheme, I present an analysis of critical questions which provide pointers to question and/or attack the plan proposal.

The remainder of this chapter is structured as follows: Section 3.1 presents an approach to provide agents with mechanisms to argue about cooperative plans. Appendix A presents an example where agents need to propose an agree on a plan in a war zone. The example will help to exemplify the elements presented throughout the thesis. Section 3.2 presents an argumentation scheme for durative action proposals. Section 3.3 present a discussion on multi-agent temporal plans and the implications of having individual action dependency in multi-agent plans. Section 3.4 presents the argumentation scheme for plan proposals: ASP. Section 3.5 presents an analysis of critical questions that match the ASP argumentation scheme where questions are categorized into seven layers. Finally, Section 3.6 presents a chapter summary.

3.1 An Approach to Propose Cooperative Plans using Argumentation Schemes

Argumentation-based dialogues (cf. Section 2.2.3) can be used to specify how agents can engage in a dialogue about plans. The approach presented in this thesis assumes agents need to propose, justify and evaluate multi-plans taken from a private plan repository (i.e. plans created by or supplied to the agents) and then engage in a dialogue to justify, evaluate and agree on a plan to execute.

Planning in multi-agent scenarios is a complex process. This thesis propose the use of a structured dialogue using argumentation principles in order to agree on a plan in situations where agents have different views about the world and/or different interests. Agents may have different information about the world because they cannot update continuously their beliefs or because they have a partial view of it. Whatever is the reason, the aim is to deal with these conflicts through dialogue.

A structured argumentation-based communication process where agents can apply their preferences in the selection of actions and plans can help to solve problems related to the *selection* and *modification* of multi-agent plans. Agents should be able then to propose and evaluate plans in a structured way while obtaining information about all the possible factors that may affect the plan execution. It is out of the scope of the thesis all the issues related to the *execution* of the plan.

Figure 3.1 an approach to plan creation where agents merge their Knowledge Bases and then create a plan. Figure 3.2 presents the approach considered in this thesis where agents discuss plans already formed and then engage in an argumentative dialogue to agree on which plan to execute. I assume that all the agents involved in the dialogue are willing to cooperate to achieve the goal and that they may have differences on their beliefs about the world and their preferences over the values promoted by the plan. An important difference between these two approaches is that agents do not merge their complete Knowledge Bases, but only exchange such information as is relevant to the proposed plans. Therefore, the objective is that agents take advantage of an argumentation-based dialogue structure to resolve their differences and provide them with options where they can use their preferences as the dialogue progresses. An agent *ag* for the purpose of this thesis has the following elements:

- a set of state predicates used to represent the world;
- a belief assignment to each of the predicates that represents the agent's beliefs about the current state of the world. The beliefs of agents do not have any uncertainty. They can be wrong about their beliefs, but they are determined;
- a set of durative actions Ac_{ag} the agent can perform with the elements that conform the action i.e. the preconditions, start effects, invariant conditions, termination conditions and end effects;

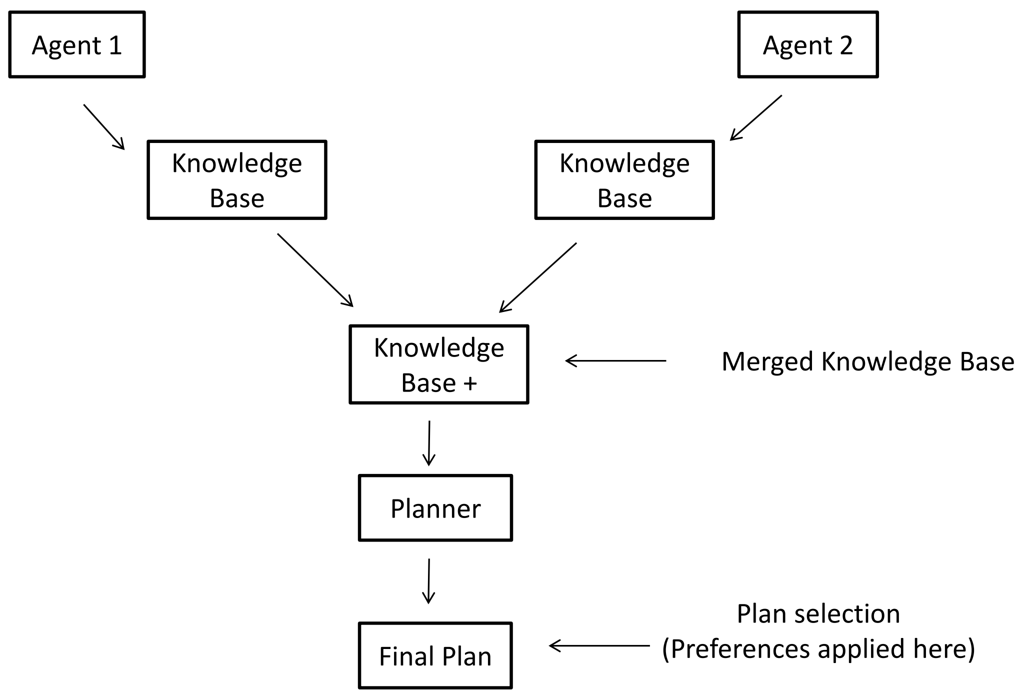


FIGURE 3.1: A Distributed Artificial Intelligence Approach to Plan Creation.

- a condition or effect of an action Ac is defined for an action if the predicate the condition is part of the set of conditions in action Ac. This will be furter explained in the next subsections;
- a *plan goal* G;
- a set of *values* V_{ag} and a preference over these values δ;
- an element indicating if the agent has evidence to support the element (e.g. a predicate, the belief assignment, an action, a value);
- an *internal planner* that receives as input the initial state of the world and a set of actions and outputs a multi-agent plan that consist of set of scheduled actions for each agent involved in it that achieve the goal;
- a set *possible* of multi-agent plans PL generated by an internal planner that achieve the goal and promotes/demotes values in V_{ag};
- a *plan checker* that receives as input the initial state of the world , a set of actions, a multi-agent plan PL and the goal G and verifies if the plan PL achieves the goal G.

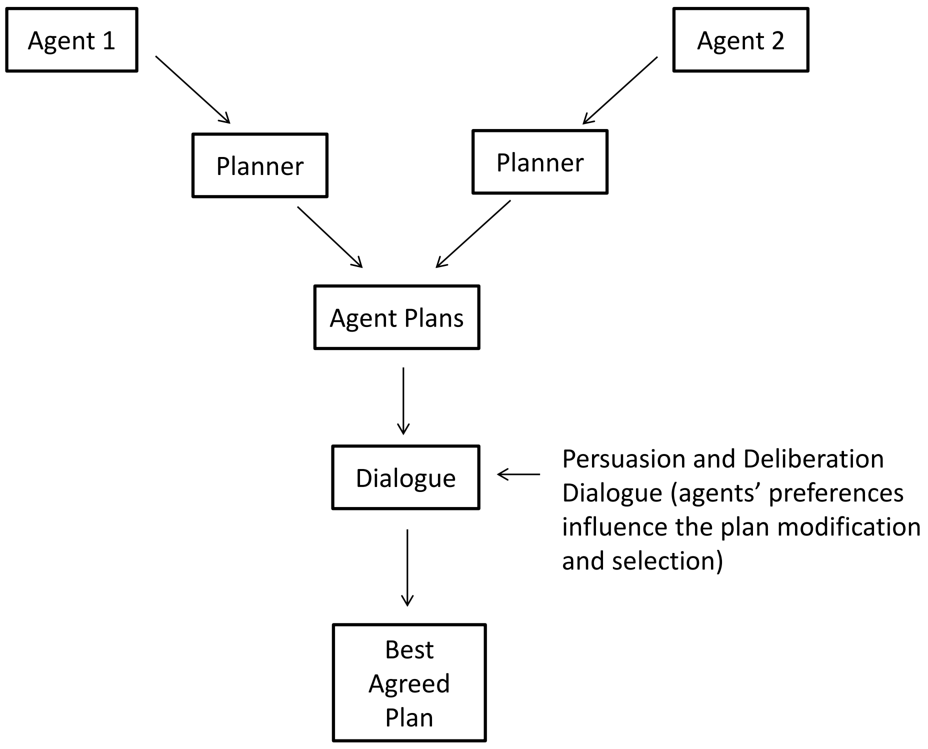

Agent 1
Agent 2
Planner
Planner
Agent Plans
Dialogue
Persuasion and Deliberation Dialogue (agents' preferences influence the plan modification and selection)
Best Agreed Plan

FIGURE 3.2: Argumentative Approach to Plan Creation.

In Appendix A, an example is presented to explain the elements introduced in the thesis. The next section presents the concept of durative actions in the context of agents proposing multi-agents plans for execution.

3.2 An Argumentation Scheme for Durative Action Proposals

The argumentation scheme for *plan proposals* presented later in this chapter builds on the argumentation scheme for *action proposals* $AS1$ presented in [11] (see Section 2.3.4). In this chapter, I present an extension to $AS1$ that considers **durative actions**.

In [11], actions in the $AS1$ argumentation scheme are presented as operations from a state or current circumstances that cause new circumstances or effects to become true resulting in a new state. The authors assume a finite set of distinct actions giving no consideration to time (i.e. the duration of the action, the start-time and end-time) and effects and conditions that hold throughout the action duration (start effects, invariant conditions and termination conditions), all of which are important to define temporal multi-agent plans.

This section takes concepts from the Planning Domain Description Language (PDDL) 2.1 temporal action specification presented in [71] used to describe actions used for temporal planning (cf. section 2.4.1). In the PDDL 2.1 action specification, instead of considering actions only with preconditions and effects, actions are presented as *durative entities* with additional conditions and effects that hold and become true at different times.

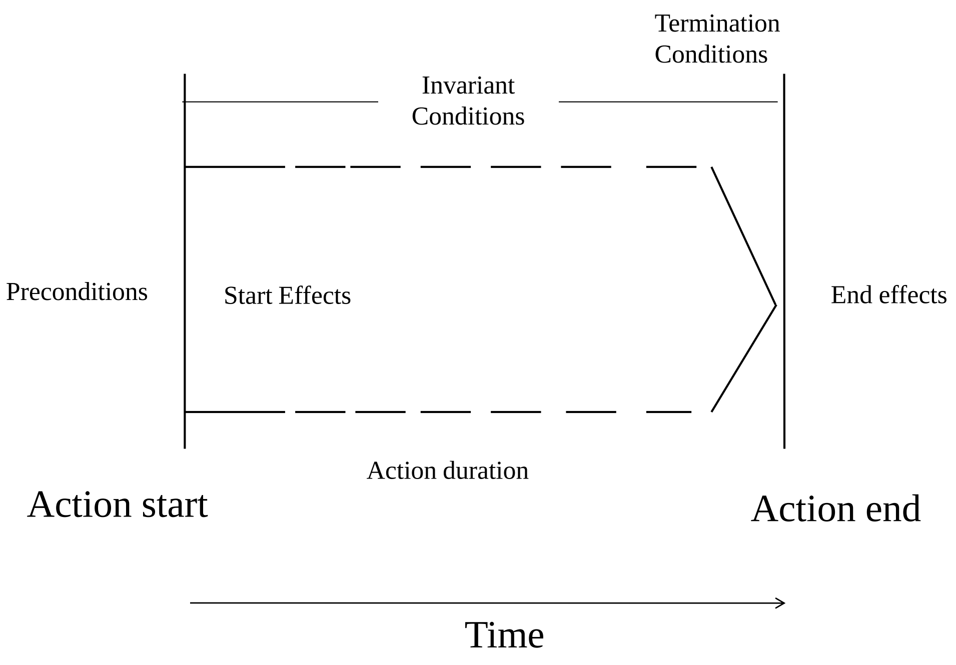


FIGURE 3.3: A PDDL 2.1 Durative Action Representation.

Figure 3.3 presents a durative action diagram as presented in the PDDL 2.1 specification. A durative action for the purpose of this thesis has the following elements and characteristics:

- A set of **preconditions** that hold before the action can start. A state is formed by a set of predicates and the *preconditions* of an action are a subset of these predicates with specific values for the action to be executed.
- The action has a **duration** or interval of execution.
- The action has a **range** which indicates the the permitted value by which an action duration can increase or decrease without affecting the effects of the action (see explanation below).
- Once the action starts a set of **start effects** becomes true. The *start effects* are predicates with specific values that are made true by the action and can be different from the end effects.
- Throughout the complete duration of the action **invariant conditions** hold. These conditions are not caused by the action.
- A set of **termination conditions** hold before the action can end.
- A set of **end effects** which begin to hold when the action is complete i.e. the effects of actions only change the value of predicates. It is assumed that predicates not mentioned in the effects of the action stay with the same value, unless another action changes their value.

The actions have a duration that may change within a *range*, that is, increasing or decreasing the duration within the range without affecting the effects of the action. From the running example, moving through zones with the action $move()$ has duration 4 with range 1, which means the action can take 5 or 3 time units without changing the effects of the action. I assume here agents have control about the duration the action, leaving aside variations in the duration due to conditions unknown to the agents. In [34], for example, the Multi-agent Planning Language MAPL (an extension of PDDL 2.1) introduces actions whose duration is determined in runtime. The intention behind of presenting durative actions with a range is to give agents elements to engage in dialogues using action elements that include temporal elements and so to enable a richer argumentation-based dialogue over cooperative plans.
An example of a durative action for the agent NGO using the example from Appendix A is as follows (see Table A.5):

the durative action $move(ag, X, Y)$ with duration 4 and a range of 1 (see Tables **??** and has *preconditions*:

(i) the agent ag should be in zone X,

(ii) the fuel should be full/low,

(iii) the route (X,Y) should exist,

(iv) Zone X should be a secure zone.

(v) Route (X,Y) should be a secure route.

The *start effects* of the action are:

(i) the fuel starts decreasing, and

(ii) the agent starts moving and is no longer in zone X.

The *invariant condition* is that the fuel keeps reducing.

The action has no *termination conditions*

The *end effects* of the action are are:

(i) the agent arrives in zone Y and

(ii) the fuel is reduced to low/empty.

Another example of a durative action is the action *control*() in Table A.9. The invariant condition for the action *control*() is that the *GroundForce* is *deployed.* This does not change throughout the action duration and is not part of the end effects. Note that $GroundForce(deployed)$ is also a start effect of the action. In this scenario it is possible to represent that an effect of the action should remain true throughout the execution of it and the way to do it is specify them as a start effect and a condition.

This model of time still presents some limitations as durative actions should allow effects and/or conditions to be asserted at arbitrary points during the interval of execution, or to be a function of duration (*until* actions). The planning community is still developing ways to create planners that handle temporally extended actions [71].

The extended argument scheme for action proposal *AS2* is as follows [1]:

[1]In [11], authors make no difference between resolving on a future action and justifying a past action, for the purpose of this thesis I assume the proposal presumptively justifies a future action part of a plan to execute.

- Given current circumstances R
- where preconditions P hold
- agent Ag_i should perform action Ac with duration d at time t_s
- that generates start effects E
- with invariant conditions I
- and termination conditions C
- with end effects EE that hold in state S
- which will realize goal G
- and promote/demote value v.

where R is the initial state, Ac is an arbitrary action of agent Ag_i, S is the final state, the goal G is a finite set of formulae that holds in S, and v is a value that justifies the transition from state R to state S. It is assumed that values are qualitative social interests of agents and the preference over these values is subjective to the agents.

An example of how the $AS2$ can be instantiated is:

- Given state R
- where preconditions: $(inZone(NGO, 1), Fuel(full))$ hold
- agent Ag_i should perform action $Ac = move(NGO, 1, 2)$ with duration d=3 at time t_s =0,
- that generates start effects: $Fuel(decreasing)$, $moving(NGO)$
- with invariant condition: $Fuel(decreasing)$
- with end effects EE $inZone(NGO, 2)$, $Fuel(low)$ that hold in state S
- which will realize goal $G = inZone(NGO, 2)$
- and promote the value v_{mob}

I now present a discussion analysis on multi-agent temporal plans, action dependency and concurrent actions and the implications to the plan proposal scheme.

3.3 Multi-agent Plans and Concurrent Actions

In the example scenario presented in Appendix A, agents have to agree on plans that are dependent of other agents' actions (e.g. agent NGO cannot arrive to zone 14 alone because it has to be escorted by agent PKF in several routes). The problem of modelling and handling the effects and interactions of multiple concurrent actions has been considered a challenging task in the Artificial Intelligence Planning community (see [155] for an account on action concurrency).

Multi-agent planning involves coordinating the actions of multiple agents towards a common goal. The central issue in multi-agent planning lies in the fact that individual agent actions can interact at some point [32]. Multi-agent plans then can be represented as individual plans for each actor with an explicit representation of the effects and interactions of multiple concurrent actions, see e.g. [32, 94][2].

Multi-Agent Temporal Planning involves reasoning about other agents actions and its duration, taking into account possible interferences. A *multi-agent temporal plan* has synchronized and scheduled concurrent actions with temporal constraints [34]. Another definition related to temporal planning is given in [52] where *metric temporal planning* is defined as: *the problem of selecting, ordering and scheduling durative actions in order to achieve a goal.* In particular, the selection of actions is a *planning* problem and the assignment of a start time to actions refers to the problem of *scheduling.*

[2]In [32] for example, concurrent actions are part of a *concurrent list* of actions available to create plans.

A *multi-agent temporal plan* then requires to take into account interactions between individual actions to handle the concurrency of actions at specific times in an efficient way [45][3].

Dependent concurrency plays a critical role in temporal planning. In [45], the authors distinguish two ways in which concurrency in multi-agent plans can arise: the first one is the concurrency derived from the interactions between the actions in the domain and the second one is the concurrency derived from a *deadline restriction* that forces actions to be compressed in time.

In [83], the authors discuss that in domains with deadline-free problems planners should be able to have a closer integration of planning and scheduling decisions in order to avoid or exploit action interaction.

Some issues regarding multi-agent planning and action concurrency are (discussed in [34] for the multi-agent planning framework MAPL):

- In multi-agent planning, concurrency can appear at the action level in two forms: (i) concurrent actions in an individual plan or (ii) as actions distributed over several plans by different agents.
- Metric time is needed to realistically describe action durations and their relations, otherwise the exact time when actions star become a problem.
- Synchronizing actions of unknown duration demands qualitative use of time.
- Synchronization on communicative acts to coordinate actions.
- The capability for single-agent to create multi-agent plans. When an agent does not know the capabilities of other agents and cannot create a valid plan to reach the goal cooperation between agents is needed e.g. asking for specific information about the agents capabilities. Coordination is also necessary when individually plans conflict with each other

This thesis assumes agents can produce temporal multi-agent plans where the durative actions between agents can depend at some points on one another, i.e. the action dependency is derived from the interactions between individual agent actions.

The concurrency of actions here should not be interpreted as *joint actions* as presented in [178] (see Section 2.3.5.1) where a joint action (without an explicit representation of time) is a function that represents the transition from a state to a *set* of states and the effect of an individual action is derived from the execution of the joint action (i.e both actions).

Several issues can arise if using joint actions to define multi-agents plans, for example:

- the specification of pre and post conditions of joint actions that start or end at different times

[3] In [45], for example, the authors present a temporal planner that schedules temporal concurrent actions (although this approach is not presented in a multi-agent context).

- the exact representation of plans as a sequence of joint actions containing the individual actions for each agent
- the number of possible joint actions (and their complexity) increases exponentially with the number of agents

In this thesis I do not investigate further consequences of the specification plans as joint actions, instead, I assume individual plans can depend on each other and are concurrent in the sense that the interval of execution of some actions may overlap.

Actions can concur in several ways depending on the time they start or end. The analysis of time intervals in [1] can be used to define ways in which actions can concur. The relationships are the following (depending on the nature of the domain and the nature of the actions, not all of these combinations actions may necessarily be possible.):

Assuming α and β are *time intervals*:

- α *precedes* β: This type of combination is common in plans that establish sequential actions such as in classical planning.
- α *meets* β: This combination is similar to the previous but the start point is immediately after the finish of the previous action.
- α *equals* β: This combination presents the case where two actions are performed in parallel.
- α *overlaps* β: Interval α starts before interval β ends.
- α *starts* β: Both intervals start a the same time and alpha finish first.
- α *during* β: Interval α starts and finish during the interval of β
- α *finishes* β: Both intervals end at the same time and beta start first.

The following list of *action combinations* presents how two actions can be combined in a plan using the analysis of time intervals presented (examples related to Appendix A are presented in Figure 3.4):

AC1: Action Ac precedes Ac_1 ($precedes(Ac, Ac_1)$) (Figure 3.4(a))
Example: *precedes* ($move(NGO, 6, 7)$ at t_0, $move(PKF, 7, 8)$ at t_4)

AC2: Action Ac meets Ac_1 ($meets(Ac, Ac_1)$) (Figure 3.4(b))
Example: *meets* ($move(NGO, 6, 7)$ at t_0, $move(PKF, 7, 8)$ at t_3)

AC3: Action Ac overlaps Ac_1 ($overlaps(Ac, Ac_1)$) (Figure 3.4(c))
Example: *overlaps* ($control(PKF)$ at t_0, $prepareResources(NGO)$ at t_4)

AC4: Action Ac starts Ac_1 ($starts(Ac, Ac_1)$) (Figure 3.4(d))
Example: *starts*($prepareResources(NGO, 6)$ at time t_1, $move(PKF, 6, 7)$ at t_1)

AC5: Action Ac is executed during action Ac_1 ($during(Ac, Ac_1)$) (Figure 3.4(e))
Example: *during* ($prepareResources(NGO, 7)$ at t_1, $control(PKF, 8)$ at t_2)

AC5: Action Ac finishes Ac_1 ($finishes(Ac, Ac_1)$) (Figure 3.4(f))
Example: *finishes* ($prepareResources(NGO, 7)$ at t_1, $control(PKF, 8)$ at t_3)

AC6: Action Ac equals Ac_1 ($equals(Ac, Ac_1)$) (Figure 3.4(g))
Example: *equals* ($checkWeapons(PKF, 5)$ at t_3, $prepareResources(NGO, 5)$ at t_3)

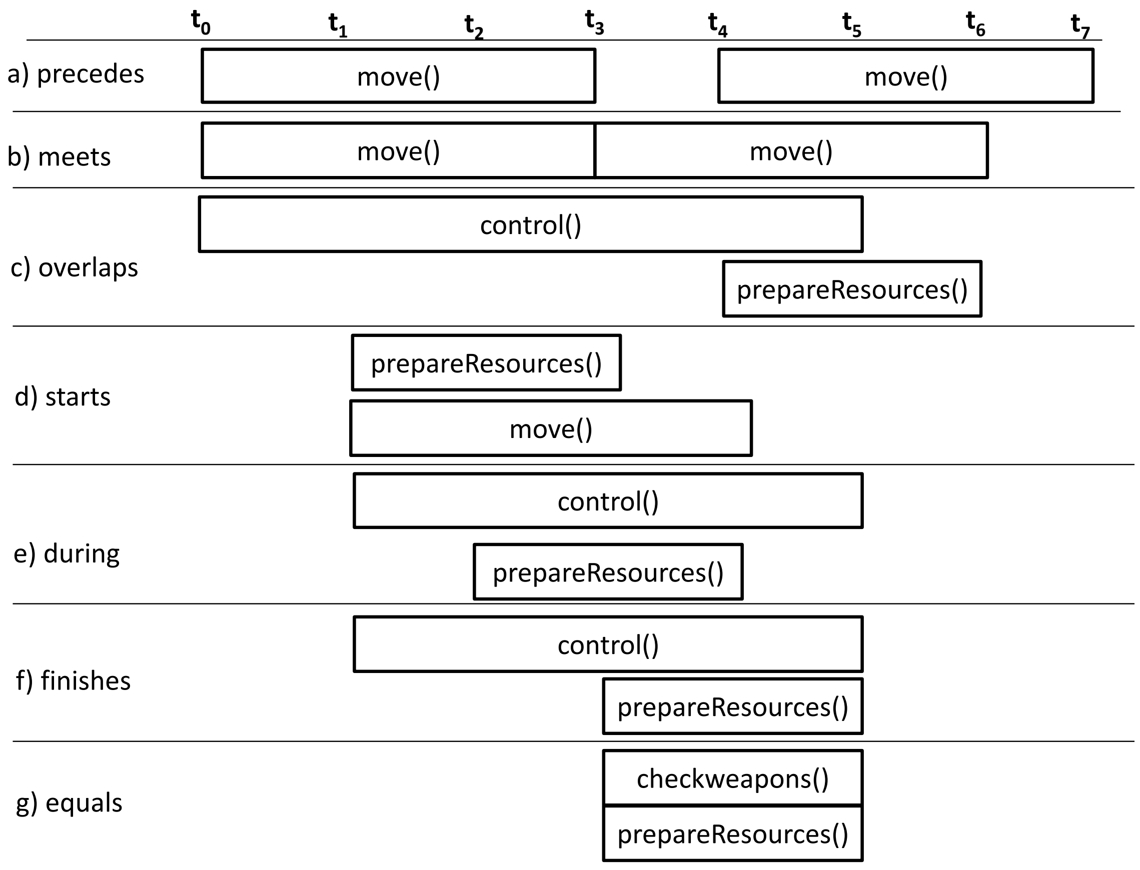


FIGURE 3.4: Examples of action combinations for the NGO Force example.

3.4 An Argumentation Scheme for Plan Proposals

This section presents the Argumentation Scheme for Plan proposals, ASP (see Table 3.1). A **multi-agent plan** is defined as a set of temporally ordered actions that details the individual sequence of actions for each agent involved and the temporal constraints on the actions.

A multi-agent plan can be defined as a tuple $PL = (R, Ag, Ac, E, S, G, V_{PL}, \delta)$ where:

- R is the initial state,
- Ag is a set of agents where $ag \in AG$
- Ac is a set of durative actions where:
  - each action has a duration d, a range r and
  - a start time Ac_t that defines the start time of the action
- E is a function that assigns to each action an agent $Ac \rightarrow Ag$
- S is the final state
- G is the plan goal
- V_{PL} is a set of values
- and δ is a valuation function $\delta : R \times S \times V_{PL} \rightarrow \{+, -, =\}$ which defines the status (promoted (+), demoted (-) or neutral (=)) of values ascribed by the agent Ag to the plan.

It is assumed that the planner input is : R, G, V_{PL} and δ and the output is plan PL.

As an example, consider the plan PL_A aAgent PKF travels directly to zones 6 via zone 4 and 5 while agent NGO waits in zone 1 and then travels directly to zone 6 via zone 2, see Table A.21):

- The initial state R: agent NGO is in zone 1, agent PKF is in zone 3 and the fuel is full for both agents.
- Ag : $\{NGO, PKF\}$
- G is the plan goal is that both agents reach zone 6.
- Ac :

$\boldsymbol{Ac_t}$	$ag : NGO$	$ag : PKF$
0	$idle(NGO, 1, d(1))$	$move(PKF, 3, 4, d(3))$
1	$idle(NGO, 1, d(1))$	
2	$idle(NGO, 1, d(1))$	
3	$move(NGO, 1, 2, d(3))$	$move(PKF, 4, 5, d(3))$
6	$idle(NGO, 2, d(1))$	$idle(NGO, 5, d(1))$
7	$move(NGO, 2, 6, d(3))$	$move(PKF, 5, 6, d(3))$

- S the final state S: agent NGO is in zone 1, agent PKF is in zone 3

- V_{PL}: v_{mov}, v_{sec}, v_{NGOsec} and v_{tt}

- δ: is v_{mov} promoted, v_{sec} promoted, v_{NGOsec} promoted and $v_{tt} promoted$

Consider now plan PL_D to reach zone 14 in Figure 3.5(see also Table A.18). In plan PL_D, NGO travels under escort of PKF from zone 6 to zone 14 via zones 7 and 10. While waiting in zone 10 for permission to enter zone 14 NGO prepares resources. Finally NGO aid in zone 14 while PKF is idle.

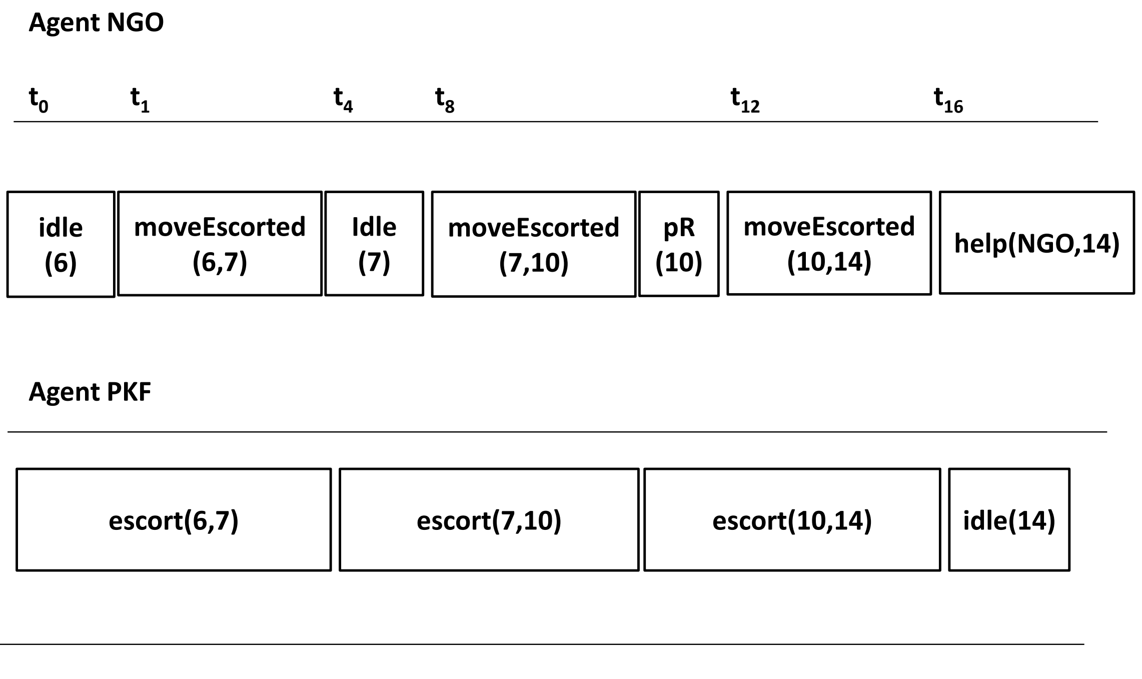


FIGURE 3.5: NGO multi-agent plan proposal PL_D to reach Zone 14 from zone 6, cf. to Table A.18.

Consider actions $moveEscorted(NGO, 6, 7)$ and $escort(PKF, 6, 7)$, note that action $escort()$ involves agent PKF going into the zone first at time t_0 with duration 4, (thus the need of the $idle()$ action for agent NGO) and then action $moveEscorted()$ with duration 3 at t_1. Action $escort()$ then *finishes* action $moveEscorted(NGO)$ (both finishing at the same time).

If we refer to these actions as just an action combination (*finishes*) there is the problem of specifying the exact point in time when $moveEscorted()$ starts. In this case, is when the precondition $GroundForce(vigilant)$ (an start effect of action $escort()$) becomes true; but this assumption is too general and poses several problems in terms of specifying multi-agent plans as sets of joint actions (see the discussion at the beginning of this section).

I assume the actions are already scheduled in the plan but agents can question the scheduling based on the type of action combinations presented or the dependencies between actions. In section 3.5.4, I present critical questions that consider the way actions from different agents can overlap.

The Argumentation Scheme for Plan proposals (ASP) is presented in Table 3.1. A valid instantiation of the scheme assumes the existence of a regulatory environment or a *social context* comprised of norms or constraints. The *social context* is an extension introduced in [13] to the argumentation scheme presented in [11] about action proposals where agents use a social structure to issue valid commands between them for example, to reflect a hierarchical structure of authority that may exist between agents. I acknowledge the importance of a *social context* or a regulatory environment in planning tasks and this is the reason to include it in the scheme, but in this thesis the ways in which norms or constraints influence the model are not analyzed in full.

The current circumstances are given by state R which is considered to be initial state where agents in Ag should perform the plan PL defined as an ordered set of durative and scheduled actions. The transition leads to state S which realizes the plan goal G and promotes values V_{PL}.

The reason behind specifying a set of values V_{PL} rather than a single value is because a plan may promote several values since it could be formed of several actions. For example, if a plan has two actions, value v_1 can be promoted by the first action of a plan and value v_2 can be promoted by the second action. So, the set of values promoted by the plan is the set of values promoted by all the actions that comprise the plan (so what is promoted by the plan as a whole may differ from what is promoted by the individual actions within it). Furthermore, a plan thus can promote multiple values and each action in the plan may promote multiple values. Essentially, the relationship between the values promoted/demoted by a plan and those promoted/demoted by its component action elements is just the union of those values.

This way to specify the promotion/demotion of values in actions and plans could indeed be extended to allow a more complex relation of value representation for the set of actions, for example: there can be values promoted during an action, the same value could be promoted and demoted at the same time in the plan with different actions, etc (from the running example take the case where agents move through secure and insecure routes promoting and demoting the value of security). In Section 8.3, a more detailed account on the possible research paths related to the representation of values in a plan is presented.

TABLE 3.1: Argumentation Scheme for Plan Proposals ASP

Given a social context,

in the current circumstances R,

agents Ag should perform multi-agent plan PL

where PL is a set of scheduled durative actions for each agent

to achieve new circumstances S

which will realize the plan goal G

and will promote values V_{PL}.

An example of how the plan proposal could be instantiated is presented below using the example from Appendix A. The plan PL_C presented in Table A.21 is presented detailing the actions of both agents and the values promoted by the plan given below (PKF travels directly to zone 6 via zone 4 and 5. NGO travels directly to zone 6 via zone 2 and waits for PKF in zone 6):

- Given the goal that agents NGO and PKF need to reach zone 6
- agent NGO has the authority to propose plan PL_C (given by the **social context**),
- NGO is in zone 1 and believes agent PKF is in zone 3
- so agents NGO and PKF need to perform together **plan** PL_C
- consisting of actions:

  $(move(NGO, 1, 2), move(PKF, 3, 4))$ at time t_0

  $(move(NGO, 2, 6), idle(PKF, 4)$ at time t_3

  $(idle(NGO, 2), move(PKF, 4, 5))$ at time t_6

  $(idle(NGO, 6), move(PKF, 5, 6))$ at time t_9
- to reach zone 6,
- and promote values v_{sec} and v_{NGOsec}

The next section presents a set of critical questions that are associated with the argumentation scheme for plan proposals ASP.

3.5 Critical Questions for Plan Proposals

In Walton [190], argumentation schemes are associated with a set of characteristic *critical questions* (cf. Section 2.3.2) so that each argumentation scheme has its own set of critical questions that matching the scheme. Specifically, critical questions represent a way to evaluate arguments constructed using an argumentation scheme and provide pointers to factors which could potentially make the argumentation scheme inapplicable.

These critical questions are designed to ensure that there is no reason to reject the presumptive conclusion of the argumentation scheme, and so each question can be seen as a potential attack on the argument presented. Critical questions provide *examination patterns* and a question could be seen as a passive attack on the argument; passive in that they do not present counter-arguments, but can provide reasons to reject the argument they critique.

The set of critical questions (in the next subsections) that match the argumentation scheme ASP is classified into the following seven layers (also presented in Figure 3.6):

Layer 1: An action and its elements (Lowest level): Questions in this layer inquire about elements in the durative action.

Layer 2: The timing of an action: Questions about the duration and timing of a durative action.

Layer 3: The way actions are combined: Questions about the way two actions can be combined in a multi-agent plan.

Layer 4: The plan proposal: Questions about the plan proposal as a whole.

Layer 5: The timing of the plan proposal: Questions about the timing of the plan.

Layer 6: Side effects of actions or the plan: A *side effect* is an unintended outcome of an action, and could in principle, promote or demote a value in contradiction to an agent's interest. Questions in this layer aim to point out side effects that can make the plan inapplicable.

Layer 7: Alternative paths (Highest Level): Questions in this layer inquire about alternative (possibly better) actions or plans.

Each layer groups questions according to the level of detail on which they focus. At the plan proposal level for example, the critical questions are all those that are independent of the way in which actions are composed inside the plan i.e. the way in which actions are combined. This classification allows us to consider questions at each layer separately in a dialogue.

The syntax, semantics, and pragmatics of questions have long been studied in the context of Linguistics [79]. The analysis of questions presented here focuses on the form

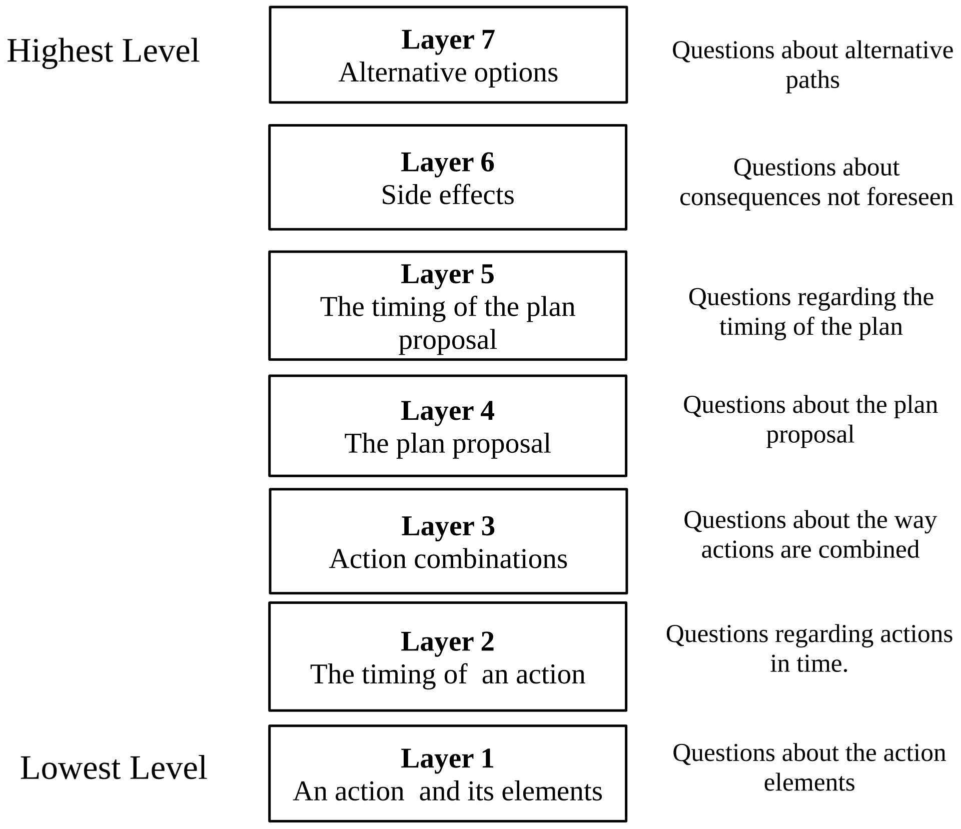

Highest Level
Layer 7
Alternative options
Questions about alternative paths
Layer 6
Side effects
Questions about consequences not foreseen
Layer 5
The timing of the plan proposal
Questions regarding the timing of the plan
Layer 4
The plan proposal
Questions about the plan proposal
Layer 3
Action combinations
Questions about the way actions are combined
Layer 2
The timing of an action
Questions regarding actions in time.
Lowest Level
Layer 1
An action and its elements
Questions about the action elements

FIGURE 3.6: Critical Question Layers for the Argumentation Scheme *ASP*.

(syntax) and the content (semantics) of the questions. The set of critical questions presented in the following subsections builds on the critical questions developed for action proposals in [11] and it is extended with the elements presented in this chapter. Thus, this analysis enables plan proposals to be questioned in a comprehensive way in order to be explicitly justified.

3.5.1 Overview of Critical Questions

According to [8], practical reasoning is a process where agents need to consider three types of questions:

- *Problem formulation*: what facts, values, interests and aspirations are relevant for the situation e.g:
  - What elements (actions,conditions,effects) are *defined* for the agent.
  - One agent may consider that the value of a predicate is different from another agent predicate definition i.e. a condition or an effect is different in the local agent's action specification from what is presented in the proposed plan.
  - Causal theory i.e. what states are reached by a given actions (the effects of the action).

  - What values are recognised and which transitions promote or demote the value.

- *Epistemic reasoning*: what is the current situation with respect to the structure formed at the previous stage;
- *Action selection*: related to the action selection based on the social values promoted by those actions.

Given these classification of questions I present here another classification of critical questions:

1. Critical questions that challenge the **validity** of an element
2. Critical questions challenge the **possibility** of an element
3. Critical questions that challenge the **possibility in time** of an element
4. Critical questions that challenge the **suitability** of an element
5. Critical questions that point out other possible better **alternatives**
6. Critical questions that inquire about **additional information**

In the context of this thesis, questions about the **validity** of an element (e.g. an action, a condition, an effect) refer to the *existence* of the element in the beliefs of the agent [4]. For example, an action Ac is *invalid* for agent NGO if PKF presents action Ac and it is not defined in the set of possible actions that PKF can perform. Conditions and effects are then *valid* predicates for an action Ac, iff the predicate is part of the set of predicates in agents action Ac specification.

This problem can be viewed as an ontology alignment problem. Thus, questions about the *validity* of elements can be seen as a part of the *problem formulation* practical reasoning stage. Further explanation will be given with the actual questions in the next subsections.

A critical question may also inquire about the ***possibility*** of an element within the scheme. When challenging the possibility of an element it is assumed that the element is valid but not possible because the specification of the element is different. Conditions and effects are *possible* to an agent ag_A iff the values of the arguments' predicates presented by agent ag_B are the equal to the value specified in agent ag_A action specification. Questions about the *possibility* can be seen as a part of the *problem formulation* practical reasoning stage.

Furthermore, an element may be valid and possible but not possible at a specified time which leads to questions about ***possibility in time***. Questions about *possibility*

[4]In literature on argumentation the concept of validity is used differently to that used in logic. For example in [11] agents are able to attack elements in proposals for action based on the validity/existence of an element.

in time, inquire specifically what is true in the current situation and can be seen as a part of the *epistemic reasoning* practical reasoning stage.

There are also ***suitability*** questions that can be issued when it is assumed that the action is valid and possible but may not be *appropriate* to execute. Questions about *suitability* can be seen as a part of the *problem formulation* stage.

Another way to challenge an element is to compare it with ***alternative*** (possibly better) options. If a better alternative from the perspective of the opponent is found, the critical question defeats the argument presented. Questions about *alternative options* can be seen as a part of the *action selection* practical reasoning stage.

Finally, there are questions that aim to get ***additional information*** regarding an element of the proposal, e.g. questions that consider ranges of time. Questions that inquire about *additional information* can be seen as a part of the *epistemic reasoning* practical reasoning stage.

Although the analysis of critical questions presented is structured, there is no systematic way to define the questions and this has been of much critique in the argumentation community. There is still a large bridge between human argumentation and computational argumentation and research on linguistics should play an important role if a system of such nature should be conceptualised [204]. A more detailed analysis on the nature of critical questions is given in Chapter 5 where I present a priority order in which questions can be considered in a dialogue.

The following subsections present the critical questions for each layer along with an explanation of it, the type of questions (taken from in the classification in [8]) and an example of each that refers to a particular case in the example from Appendix A.

3.5.2 Layer 1. Critical Questions for Actions.

This layer has critical questions that consider different ways to attack an individual action. I extended the list of critical questions for action proposals in [11] considering the characteristics of a durative action. The question examples in this section are asked from the NGO perspective assuming agent PKF has wrong information about the world.

CQA-01*. Is the action valid (defined)?*

This is a *problem formulation* question that inquiries about the validity of an action that is, its definition in the problem (cf. Section 3.5.1 to the notion of validity used in this thesis). Thus, an agent ag_A has reason to believe that an action Ac proposed by ag_B is *invalid* iff Ac is not specified in ag_A local action specification.

Example: If agent PKF puts forward a plan with action $askforBackup(PKF)$ (see Table 3.2), the action is not defined in NGO's action definition, so the agent has reason to believe that the action $askforBackup(PKF)$ is *invalid.*

CQA-02*. Are the action preconditions valid (defined)?*
This question inquires the validity of an action precondition. An agent ag_A has reason to believe that a precondition p part of an action Ac proposed by agentag_B is *invalid* iff p is not specified in the local action specification for agent ag_A.

Example: If agent NGO presents a plan with action $move()$, the action definition includes the precondition: $startTime(Beforenoon)$, thus, agent PKF can question/challenge the use of the precondition in the action e.g. *Is the precondition $startTime(beforeNoon)$ for the action $move(NGO, 6, 7)$ a valid precondition?*. The agent PKF does not use the precondition in its definition of the action $move()$ (see Table A.5) so, agent PKF can challenge the precondition with this question.

CQA-03. *Are the action preconditions possible in the current state?*
This is an *epistemic* question that inquires about the preconditions of an action. Preconditions can be valid (defined for both agents) but not possible in the current state (cf. to Section 3.5.1) An agent ag has reason to believe a precondition is not possible iff the value of the predicate presented is different in its local definition of the world.

Example: Consider plan PL_C (see Table A.22) and the initial state: ($inZone(NGO, 1)$ and $inZone(PKF, 3)$, agent PKF presents the plan which includes action $move(NGO, 2, 6)$ at time t_0, agent NGO can challenge the precondition $inZone(NGO, 2)$ (of action $move(NGO, 2, 6)$)is not possible in the initial state since NGO location at time t_0 is $inZone(NGO, 1)$.

CQA-04. *Are the action start effects of the action valid (defined)?*

This is a *problem formulation* question. As with questions *CQA-01* and *CQA-02* id effect presented as part of an action is not defined in the local action specification of the agent the predicate can be challenged. An agent ag_A has reason to believe a start effect presented by another went is not valid iff the start effect is not specified in the agent local action definition.

Example: Let's assume agent PKF presents a plan that includes action $escort(PKF, NGO, 2, 6)$ at time t_3. For agent NGO the effect $flanks(Covered)$ is not defined as a valid effect in the local action definition for agent PKF (see Table A.7), thus, agent NGO can pose the question *Is the start effect:* $flanks(Covered)$ *defined for action* $escort(PKF, NGO, 8)$*?*

CQA-05. *Are the action start effects possible?*
This is an *epistemic* question that inquires about the start effects of an action. Start effects can be valid (defined for both agents) but not possible in the desired state. An agent ag_i has reason to believe an effect is not possible iff the value of the predicate presented is different in its local representation of the world.

Example: Consider agent NGO presents a plan that includes action $escort()$ agent PKF can challenge that the effect $groundForce(Deployed)$ is not possible since the effect in its local definition is $groundForce(Ready)$ (see Table A.7).

CQA-06. *Are the action invariants conditions valid?*

This is a *problem formulation* question that inquires for a condition not defined in the agent's action specification. An agent ag_A has reason to believe an invariant condition is not valid iff the condition is not specified in the agent local action representation for agent ag_B.

Example: Let's assume agent PKF presents a plan that includes action $help(NGO, 14)$ at time t_3 with invariant condition $zone(protected)$, thus NGO can pose the question *Is the invariant condition* $zone(protected)$ *defined for the action* $help(NGO, 14)$*?* The condition is not defined for action $help()$ in NGO action specification, see Table A.13.

CQA-07. *Are the action termination conditions valid?*

This is a *problem formulation* question that inquires for a termination condition that is not defined in the agent's action specification. An agent ag_A has reason to believe a termination condition is not valid iff the termination condition is not specified in the agent local action definition.

Example: Let's assume agent PKF presents a plan that includes action $move()$ at time t_3 with invariant condition $inZone(Truck, X)$, thus NGO can pose the question *Is the termination condition* $inZone(Truck, X)$ *defined for the action* $help(NGO, 14)$*?* The condition is not defined for action $help()$ in NGO action specification, see Table A.5.

CQA-08. *Are the action end effects valid?*

This is a *problem formulation* question. An agent ag_A has reason to believe an effect is not valid iff the effect is not specified in the agent's local action definition.

Example: Let's assume agent PKF presents a plan that includes action $move(NGO, 6, 7)$ at time t_3. For agent NGO the effect $aidResources(low)$is not defined as a valid effect in its local action definition (see Table A.5), thus, NGO can pose the question *Is the end effect:* $aidResources(Low)$ *defined for action* $move(NGO, 6, 7)$*?*

CQA-09. *Are the action end effects possible?*

This is an *epistemic* question that inquires about the effects of an action. End effects can be valid (defined for both agents) but not possible in the desired state. An agent ag_i has reason to believe an effect is not possible iff the value of a predicate presented is different in its local representation of the world.

Example: Consider agent NGO presents a plan that includes action $escort()$ agent PKF can challenge that the effect $groundForce(Deployed)$ is not possible since the effect in its local definition is $groundForce(Ready)$ (see Table A.7).

CQA-10. *Are the new circumstances already achieved?*

This is an *epistemic* question. In continually evolving contexts, new circumstances could be true before or during the execution of the plan. This critical question presents the case where an action may already have been executed or another action may have caused the new circumstances to be true. An agent ag has reason to believe the new circumstances have been already achieved iff the current state is the same as the state in new state.

Example: *Are the circumstances:* $inZone(NGO, 14)$*, already achieved?* where $inZone(NGO, 14)$ is the new circumstance that the plan tries to achieve.

CQA-11. *Is the value promoted by the execution of the action?*

This is a *problem formulation* question. This question assumes the value is legitimate but is not promoted by the execution of the action. An agent ag has reason to believe a a value is not promoted by the action iff in its local definition the value is demoted.

Example: If agent NGO presents action $move(6, 9)$ that promotes the value v_sec, agent PKF has reason to challenge the promotion with: *The value* v_{sec} *is demoted by action* $move(6, 8)$ *since route (6,9) is a warning route.* (see Table A.2).

3.5.3 Layer 2. Critical Questions for the Timing of an Action

This layer presents questions that focus on the duration and timing of an action abstracting other details about the action. The questions challenge the start-time, end-time and duration of the action with different variations. All the questions in this layer refer to the problem formulation. Any change in the duration or start-time of the actions implies a plan revision or plan check task. This questions then provide pointers in which a check task is needed.

CQAT-01. *Is the action possible with the specified duration?*

This question challenges the possibility of the action related to its duration. A conflict in the action duration specification between agents can lead to discard the action. An agent ag has reason to believe an action is not possible with the specified duration if in its local specification the action duration has a different value.

Example: Consider a situation where agent PKF presents action $moveEscorted(NGO, 6, 8)$ with duration 2 (see Table **??**) in a plan. Agent NGO can question the action duration since its local definition of the action $moveEscorted()$ the action duration has value 3 (see Tables A.4 and **??**).

The action $moveEscorted(NGO, 6, 8)$ *is not possible with duration 3.*

CQAT-02. *Can the action duration be less?*

This question aims to change the duration of the action by reducing the duration of it. An agent ag has reason to believe the duration of an action can be *less* iff: (1) the action has a range different to 0, and (2) the proposed new duration is at least the original duration minus the range, and (3) the change does not prevent the goal to be achieved. An agent knows that reducing the duration of the action does not prevent the goal to be achieved by performing a plan check (cf. Section 3.1, the agent has a plan checker that verifies the plan achieves the goal).

Example: Consider plan PL_F where agent NGO propose action $prepareResources()$ with duration 2 at time t_4. Agent PKF can suggest to reduce the duration of the action (since the range of the action allows it) to a duration of 1 and start action $escort(PKF, NGO, 7, 8, d(4))$ at t_4 and save time.

Can the action $prepareResources(NGO, 14)$ *have duration 1?*

CQAT-03. *Can the action duration be longer?*

This question aims to change the duration of the action by extending the duration of it. An agent ag has reason to believe the duration of action Ac can increase iff (1) the action has a range different to 0 and (2) the proposed new duration is at least the original duration plus the range and (3) the change does not prevent the goal to be achieved.

Consider plan PL_F where agent NGO propose action $escort(PKF, NGO, 6, 7, d(4))$ with duration 4 at time t_0. Agent PKF can suggest to increase the duration of the action (since the range of the action allows it) to a duration of 5 and start action $escort(PKF, NGO, 7, 8, d(4))$ at t_4. Note that this change affects the execution of action $moveEscorted()$ and thus the planner may have to reschedule the actions of NGO.

Can the action $escort(PKF, NGO, 7, 8, d(4))$ *have duration 5?.*

CQAT-04. *Is the action Ac possible at the specified start time t?*

The possibility of an action at a specific time can be challenged with this question. The action may be possible with the specified duration but not at the specified start time t. An agent ag has reason to believe the action Ac is not possible at time t iff according to the agent ag the precondition p will not hold at time t.

Example: Consider NGO's plan PL_F (see Table A.19) and the action $moveEscorted(NGO, 8, 14, d(3))$ at time t_8. The action cannot be performed at time t_8 since the precondition $groundForce(vigilant)$ will not be met since action escort needs to start before.

CQAT-05. *Can the action Ac start earlier?*

This question aims to change the start time of the action without preventing the goal to be achieved. An agent ag has reason to believe an action Ac starting at time t can start earlier iff either by (1) extending the action duration within the range or (2) moving the action to a earlier start time t_i and the goal is still achieved.

Example: Consider action $move(PKF, 4, 5, d(3))$ at t_4 in plan PL_B (see Table A.17). The action can be performed earlier at time t_3 without affecting the plan outcome.

Is it possible to change the start time of action $move(PKF, 4, 5, d(3))$ *to* t_5*?*. The action can also can be performed at t_3 with duration 5 ($move(PKF, 4, 5, d(5))$) since the range allows it. As explained at the beginning of this section the answer and creation of this question depends on a plan check.

CQAT-06. *Can the action Ac start later?*

This question aims to change the start time of the action to a later time without affecting the plan. An agent ag has reason to believe an action Ac can start later iff either by (1) extending the action duration within the action range or (2) changing the action to a later start time t_i does not prevent the goal to be achieved.

Example: Consider action $move(PKF, 4, 5, d(3))$ at t_4 in plan PL_B (see Table A.17), the action can be performed later at time t_5 without affecting the plan outcome. *Is it possible to change the start time of action* $move(PKF, 4, 5, d(3))$ *to* t_5*?*.

3.5.4 Layer 3. Critical Questions for the way actions concur

The questions in this layer challenge the way two actions from different agents can concur in a multi-agent plan and form part of the problem formulation. The way actions can depend on each other in a plan is a task of the planner, so the agents need to refer to the *plan checker* to generate (or respond) questions in this layer. Disagreements in this layer are represented as conflicts in predicates of different agents. A conflict in a predicate for agent ag happens when the value of the predicate in its local representation is different from the value agent ag_1 presents.

CQAC-01. *Can sequential actions* Ac_i *and* Ac_j *be performed concurrently at some point?*

This question aims to modify the start time of the actions in order to execute them concurrently at some point. An agent ag has reason to believe that two sequential actions Ac_1 and Ac_2 from different agents in a multi-agent plan can be performed concurrently iff (1) for all the conditions and effects in the action Ac_1, if the same predicate exists in a condition or an effect of Action Ac_2, the values of the predicates are equal, and (2) the plan goal can still achieved by the plan.
Example: Take the actions $move(NGO, 2, 6)$ at t_3 and $move(PKF, 4, 5)$ at t_6 in plan PL_C (see Table A.22) at t_0, these actions can be performed concurrently at t_3 since none of the conditions and effects of the actions are in conflict i.e. none of the predicates involved have different values and the goal can still be achieved.

CQAC-02*. Can the order of the sequential actions Ac_i and Ac_j be changed?*

This question aims to change the order in which two actions are executed. An agent ag has reason to believe that two sequential actions Ac_1 and Ac_2 from different agents in a multi-agent plan can be performed concurrently iff (1) for all the conditions and effects in the action Ac_1, if the same predicate exists in a condition or an effect of Action Ac_2, the values of the predicates are equal, and (2) the goal can still be achieved.
Example: consider plan PL_C (Table A.22), and the question *Can the order of the sequential actions $move(NGO, 2, 6)$ at t_3 and $move(NGO, 4, 5)$ at t_6 be changed?*. Action orders can be changed since the are no conflicts in the values of the conditions and effects of both actions. The effects of the actions $move(NGO, 2, 6)$ and $move(NGO, 4, 5)$ are the same when performed at t_6 and t_3 respectively.
On the other hand, in plan PL_F, the actions $escort()$ and $moveEscorted()$ at t_0 and t_1 respectively, cannot change their order since the precondition $groundForce(Vigilant)$ of action $moveEscorted()$needs to be true and is made true by action $escort()$.

CQAC-03*. Is there a conflict in any of the conditions of the concurrent actions Ac_i and Ac_j ?*

A conflict in the conditions (preconditions, invariant conditions or termination conditions) of concurrent actions can lead to dismiss an action. An agent ag has reason to believe there is conflict in the conditions of actions Ac_i and Ac_j iff at least one condition of action Ac_i is in conflict with the conditions of action Ac_j.

Example: *Is there a conflict in any of the conditions of the concurrent actions $move(NGO, 2, 6)$ and $escort(PKF, 2, 6)$?* These actions cannot be performed concurrently because of a conflict with the precondition $safeRoute(6, 8)$ (of NGO action $move(NGO, 2, 6)$) in conflict with the precondition $warningRoute(2, 6)$ (of PKF action $escort(PKF, 6, 8)$). The possible combination of conflicts makes it difficult to represent all the possible conflicts in a single question. The corresponding planner should be able to avoid these inconsistencies.

CQAC-04. *Is there a conflict in the effects of the concurrent actions Ac_i and Ac_j ?*

A conflict in the effects of concurrent actions can lead to dismiss an action. An agent ag has reason to believe there is conflict in the effects of actions Ac_i and Ac_j iff at least one effect of action Ac_i is in conflict with the effects of action Ac_j.

Example: *Is there a conflict in any of the effects of the concurrent actions $help(NGO, 14)$ and $checkWeapons(PKF, 14)$?* The actions cannot be performed concurrently since the effect $groundForce(Helping)$ in action $help()$ contradicts with the effect $groundForce(CheckingWeapons)$ for the action $checkingWeapons()$.

3.5.5 Layer 4. Critical Questions for the Plan Proposal

This layer presents questions that consider the plan as a single entity with no regard to the actions that comprise the plan. I assume the plan presented refer only to names of the actions and the action parameters are held in the agents' beliefs.

CQPP-01. *Is the plan PL possible?*

Conflicts that make the plan not possible could be at several levels. This questions represents the link between the plan and its elements. This question leads to argue at different plan levels or challenge the fact that the plan does not reach the goal. Therefore, this question can be seen as a *rebuttal* for the plan argument. An agent has reason to believe a plan is not possible if an element of plan PL is not in possible according to its internal plan checker.

Example: Consider plan PL_D (see Table A.18). The plan is not possible since agent NGO did not considered a conflict in zone 7 and zone 10. *Is the plan PL_D possible?*

CQPP-02. *Is the initial state possible?*

This is an epistemic question. If the initial state believed by an agent is different from the state presented in the proposal the agent can challenge the plan with this question. This question is similar to question *CQA-03* that challenges the preconditions of an action but this one refers to the overall initial state for both agents. An agent ag has reason to believe the initial state is not possible iff at least one of the predicates that conform the initial state has a different value in its internal definition.

Example: Consider PKF plan PL_C see Table A.22 where the initial state is ($inZone(NGO, 1)$ and $inZone(PKF, 3)$). Agent NGO can challenge the initial state with: *the initial state is not possible, since agent NGO is not in zone 1 ($inZone(NGO, 1)$) is in zone 2 $inZone(NGO, 2)$*

CQPP-03. *Does the new circumstances (final state) already pertain?*

This is a problem formulation question that inquires if the actual state is the same as the desired state. This question is related to the question *CQA-10* but it is related to the final state of the plan. An agent *ag* has reason to believe the new circumstances already pertain iff the actual state observable by the agent and the new circumstances stated are the same.

Example: *Does the final state (*$inZone(NGO, 14)$ *and* $AidForce(ready)$*) already hold?*

CQPP-04. *Assuming the plan is possible, will the plan* PL *bring about the desired goal* G*?*

This is a problem formulation question that challenges the fact that the plan does not bring about the plan goal. An agent *ag* has reason to believe a plan will not bring about the goal iff the goal is not a subset of any of the the effect of the actions in the plan, and the effects of other actions does not revert the goal.

Example: *Assuming the plan is possible, will plan* PL_{G1} *bring about goal to secure zone 14?*. In this case the modified plan PL_{G1} is a plan that is possible and realize the goal since the goal is a subset of the effects of action *help*() and no other action revert the goal.

CQPP-05. *Are the values*V_{PL} *that promote the plan legitimate values?*

This is a problem formulation question. Agents may hold different values to promote the plan. An agent has reason to believe value v_n is not legitimate iff the value specification is not in its local value definition.

Example: Let's assume agent NGO propose an plan that promotes value $v_{saveResources}$. Agent PKF has reason to question since this value is not in its value definition. *Is the value* $v_{saveResources}$ *a legitimate value in the context?*

CQPP-06. *Is the value v_n promoted by the plan PL?*

This question reveals differences on how agents use the values to promote the plan. Values can be promoted or demoted in several ways. As mentioned in Section 3.4 I assume the promotion or demotion of values is related to the plan. An agent ag has reason to believe that plan PL does not promote value v_n iff in its definition of the world the plan demotes the value

Example: Consider plan PL_G Agent NGO believes zone 9 is a "warning zone" thus the plan demotes v_{sec} and agent PKF believes zone 9 is a "secure zone" thus the plan promotes v_{sec}. Thus agent NGO can question agent PKF with *Is the value v_{sec} promoted by the plan PL_G by going through zone 10?*

CQPP-07. *Is the goal G already achieved?*

This question inquires if the goal is already achieved. An agent ag has reason to believe the goal G is achieved iff the goal is a subset of the actual state.

Example: *Has the zone in crisis already been secured?*

3.5.6 Layer 5. Critical Questions for the Timing of the Plan

In this layer the plan proposal is questioned/challenged in the same way a single action was questioned in Layer 2 but related to the timing of the plan. Questions include pointers to suggest a change in the time the plan start or finish.

CQPPT-01. *Can start time of the plan be modified?*

This question aims to change the start time of the plan. Since a multi-agent plan in this thesis is conceptualized as two individual (possible dependent) plans, one for each agent, the start time of the multi-agent plan is considered to be the earliest time an action of the plan begins. An agent ag has reason to believe the start time of the plan can modify iff the effects of the plan are not modified by the new start time and the goal is still achieved. In contexts where there exist restrictions about time in which the goal should be achieved, questions about the start time of the plan are more relevant. This question relies on the planner to perform validation task to check if the goal is still achieved with the change.

Example: *Can start time of the plan PL_E be modified?* If the start time of the plan PL_E is changed to time t_1, the modification does not have consequences to the effects of the plan and the goal is still achieved.

CQPPT-02. *Is the plan possible with the specified duration?*

The possibility of the plan with the specified duration is given by the duration of the actions that comprise the plan specified in question *CQAT-01*. An agent ag has reason to believe a plan is not possible with the specified duration iff the new circumstances cannot be achieved at the time specified. This action leads to change the duration of the actions involved within their range.

Example: *Is the plan PL_C possible with the specified duration?* This plan has duration 12. Let's assume the zones can change from an status of a "humanitarian crisis" to an status of "devastated zone" when no aid arrives in time. This fact would require agents to arrive to the zone in crisis within a specified time and plans that exceed that duration would be considered not possible.

3.5.7 Layer 6. Critical Questions for the Side Effects of the Plan.

A side effect is an outcome of the action that was unintended, and could in principle promote or demote a value. All the questions in this layer refer to the problem formulation.

CQSE-01. *Does the plan PL have a side effect which demotes the value v_{PL}?*

Side effects of the plan may interfere with agents' subjective preferences over values. This question leads to dismiss the plan based on the respondent agent's different value priorities. An agent ag has reason to believe that a plan has a side effect which demotes a value iff the effect of an action in its local definition of value causes the demotion of a value as a result of the execution of that action.

Example: Consider plan PL_G (see Table A.24) and agent NGO presenting the question *Does the plan PL_G have a side effect which demotes the value v_{sec}?* Agent NGO has reason to believe that action $moveEscorted(NGO, 6, 9, d(3))$ at time t_1 demotes value v_{sec} since zone 9 is a "warning zone" and moving through warning zones demotes the value v_{sec} (see Table A.16).

CQSE-02. *Does performing the plan have a side effect which demotes some other value v_n?*

An agent ag has reason to believe that a plan demotes a value v in a plan iff the value is demoted according its local rules for demotion of the value and it is not specified in the plan.

Example: Consider plan PL_E (see Table A.23), agent PKF can question agent NGO with *Does performing the plan PL_E have a side effect which demotes value v_{sec}?*. The demotion of value v_{sec} is not presented in the plan but it demotes value v_{sec} since zone 8 is a conflict zone.

CQSE-03. *Does performing the plan preclude doing some other action which would promote some other value v_u?*

This question is related to the local values of the agent. It is assumed that agents are cooperative but always try to put forward plans that promote their preferred values. This question refers to side effects that prevent the promotion of a desired value to the agent. An agent ag has reason to believe that performing a plan PL precludes the execution of action Ac and the promotion of the value v_u iff the action Ac is not part of the plan PL, the value v_u is not promoted by the plan, the value v_u is preferred by agent ag and the action Ac promotes value v_u.

3.5.8 Layer 7. Critical Questions inquiring about Alternative Options

Questions in this layer gives chance to the agents to present and consider other, possibly better, alternatives.

CQAO-01. *Is there an alternative plan to promote the same value v_{PL}?*

An agent *ag* has reason to believe there is an alternative plan to promote the same values iff in its local plans exists a plan PL_x that promotes the same set of values v_{PL}.

Example: Let's assume agent NGO presents plan PL_A *Is there an alternative plan to PL_A that promotes the same values?* Agent PKF has reason to present plan PL_C on the basis that promotes the same set of values (see Tables A.21 and A.22).

CQAO-02. *Is there an alternative plan to realise the same new circumstances?*

An agent *ag* has reason to believe there is an alternative plan to reach the same new circumstances iff in its local plans exists a plan PL_x that realises the same new circumstances.

Example: Agent NGO *There is an alternative plan to PL_G being plan PL_{G1} that achieves the same new circumstances in less time.*

CQAO-03. *Is there an alternative plan to realise the same goal?*

An agent *ag* has reason to believe there is an alternative plan to realise the same goal iff in its local plans exists a plan PL_x that realise the same goal.

Example: *There is an alternative plan to PL_G being plan PL_D that achieves the goal villagers in zone 14.*

3.6 Chapter Summary

This chapter presents an argumentation scheme for plan proposals together with critical questions that challenge the scheme at several levels. multi-agent planning is known to be a highly complex and detailed problem due to the need to represent and reason about a large number of elements. I described an argumentation-based approach that captures how multi-agent plans can be criticized and therefore justified using a large number of questions.

The practical reasoning scheme in [9] was extended with durative actions to present argumentation scheme for plan proposals. Although [9] may reason about plans as single monolithic actions, the extension presented allow us to consider thoroughly the plans taking into account their components. To specify multi-agent plans I consider individual temporal plans that have temporal constraints between them. An analysis action combinations also has been presented to in order to tease out the ways in which actions can concur in a multi-agent plan.

The set of critical questions is categorized in seven layers that focus on different aspects of the proposal going from the lowest level which refers to actions, to the highest level which inquires about alternative options. A common remark when using critical questions as a way to challenge argumentation schemes is how to be sure that the full set of possible questions has been considered. Whilst I leave open the possibility for further questions to be added to the categories, I have generated the list from a systematic analysis of the various elements of the argumentation scheme and hence believe that it can be taken as complete for the current purposes. The critical questions address each element of a proposed plan and so they are comprehensive with respect to the representation I have chosen for plan proposals. Furthermore, the reason to classify questions suggests that another layer could be added to the list or more questions could be added to a specific layer to complete the analysis for more specific purposes.

The contribution of this chapter in relation to the thesis is that it enables a plan proposal to be considered automatically by software agents engaged in a dialogue over plans. Another important contribution is that I have articulated a novel list of critical questions related to an argumentation scheme for plan proposals that includes temporal aspects. The argumentation scheme presented is as part of the dialogue game protocol in the next chapter in which agents can propose plans and critique the proposals through specific locutions using the questions presented in this chapter.

Chapter 4

A Planning Dialogue Game Protocol

In the previous chapter I introduced an argumentation scheme for plan proposals where plans are conceptualized as scheduled combinations of actions and an analysis of the critical questions that match the scheme at several levels. In this chapter, I present the **P**lanning **D**ialogue **G**ame **P**rotocol (***PDGP***), that allows agents to evaluate plans using the critical questions from the previous chapter. Agents can use then the protocol to engage in a dialogue to propose, critique and modify plans using elements from persuasion and deliberation dialogue protocols. The protocol allows the proposal to be discussed at the action level using the same argumentation scheme and critical questions, so enabling the proposal to be discussed at all of the levels identified.

I explained in Section 2.4.3 the importance of having a mixed-type protocol to enable a comprehensive argumentative dialogue about plans. The *PDGP* protocol is based on persuasion and deliberation protocols deigned for agent communication but could be seen as a "Discovery Dialogue" (as in [107]) where the goal is to choose the best plan given that agents need to find an explanation or justification of plans in terms of the actions that comprise them.

The remainder of this chapter is structured as follows: Section 4.1 presents the underlying theory of the *PDGP* protocol that enables the critique and modification of plans. The Section is structured as follows: Subsection 4.1.1 presents the *roles* agents can take when using the protocol and the *dialogue stages* that comprise the protocol and subsection 4.1.2 presents the main rules of the protocol. Subsection 4.1.3 presents the *syntax* of the protocol. Subsection 4.1.4 presents the *semantics* of the protocol in terms of preconditions and postconditions. Section 4.2 presents a protocol evaluation based on the desiderata for dialogue argumentation protocols in [112]. Section 4.3 presents two example dialogues using the protocol based on the example from Appendix A. Finally, Section 4.4 presents the chapter conclusion.

4.1 *PDGP*: A Planning Dialogue Game Protocol

In argumentation-based dialogues information is shared using formalisms that specify the rules to exchange locutions between agents. Agent interaction based on dialectical argumentation consists of agents presenting arguments for and against a claim that conforms to *protocol rules*. Specifically, an agent interaction protocol provides valid locutions and the semantics of these locutions so that agents can present arguments to enable communication (a discussion on the transition from agent communication languages to protocol-based communication is presented in Section 2.2.3).

The *PDGP* protocol embodies a theory of justification for plan proposals that allows agents to agree on a plan to execute following the argumentation theory from Walton [186, 187, 190]. The *PDGP* protocol takes elements from on three known argumentation-based protocols (all of which were discussed in Section 2.2.3.5):

- the *Fatio* protocol [110], a protocol that permits agents to make, question, challenge and justify assertions, using an explicit argumentation theory,
- the *PARMA* protocol [10], a dialogue game protocol for action proposals and the
- the Command Dialogue Protocol *CDP* [13], a dialogue protocol for command dialogues.

In [58, 132], is presented a discussion on the the importance of having agents able to handle mixed types of dialogues in cooperative scenarios. When agents need to present proposals for plans, evaluate them and possibly modify them, several types of dialogue could be used. Planning scenarios like the one presented in Appendix A can be very complex and can require in principle several types of dialogue. Agents may try to *persuade* each other to agree on a plan or action, or may *deliberate* on the best action given a point where there are only invalid plans. To create a plan *information-seeking* dialogues would also be required in order to retrieve more information from other agents to create the plan or agents could *negotiate* over the plan resources.

The *PDGP* takes a persuasion and deliberation approach. The purpose of creating a protocol that enables first *persuasion* and possibly *deliberation* is to allow agents to persuade each other with their already created plans and modify plans if necessary avoiding a complete re-planning process. When agents identify the action that causes a conflict agents can re-plan or deliberate over the next course of action from a point in time. In *Persuasion Dialogues* participants ask for and provide reasons for their claims; in consequence, the information agreed by the participants changes during a dialogue and the goal of a persuasion dialogue is to resolve a conflict of opinion[1] [142]. The protocol presented thus aims to fulfil a specific need arises from the issues when critiquing plans and it is by no means comprehensive for all the possible types of dialogue necessary for a comprehensive dialogue about plans in terms of creation and execution (see Section

[1]Walton's *The New Dialectic* [187], suggests persuasion is only about beliefs but in later work Walton talks about persuasion also being over actions [12].

2.4.3 for a list of dialogue variations about plans). The next subsections presents in detail teh characteristics of the protocol.

4.1.1 *PDGP* Agent Roles and Stages

The protocol allows three roles that participants can take in the dialogue:

- the *Participant* role: All the agents that enter the dialogue, the role allows them to put forward plan proposals.
- the *Proponent* role: Once an agent propose a plan
- the *Respondent* role: An agent that questions or challenges the proposal role and can challenge the elements of the proposal engaging in a question/answer dialectic with the *Proponent* forcing him to provide evidence to justifying the elements of the proposal.

When a *Proponent* agents retracts a proposal its role and the role of the *Respondent* agent returns to be *Participant.* Agents cannot hold two roles at the same time. All the agents that enter the dialogue are *participants* but once an agent puts forward a proposal its role changes to *Proponent* preventing him from question his own proposal.

The *protocol stages* are based on the eight stages presented in [106] where dialogue stages for a deliberation dialogue are specified as part of a formal framework. In [89], Hulstijn uses a similar model for negotiation dialogues. Table 4.1 presents the stages considered by Hulstijn in [89], McBurney et al. in [106] and the *PDGP* stages.

The "*Inform*" stage in [89] and [106] aims to seek information related to the negotiation options and goals to be achieved respectively. The *PDGP* assumes the goal of the plan is already set and cannot change and consequently omits the "*Inform*" stage. The "*Consider*" stage from [106] corresponds to the "*Evaluation*" stage in the *PDGP* protocol and the "*Revise*" stage correspond to the "*Refinement*" stage. Finally, the "*Recommend*" stage is similar to the "*Selection*" stage in the *PDGP*. Figure 4.2 presents the protocol as a state transition diagram (omitting the *Change* stage). The syntax of the locutions is presented in Subsection 4.1.3. I now describe in detail the seven stages of the *PDGP* protocol:

1. **Opening stage**: This stage is to register all the dialogue participants. I assume all agents are cooperative and have already agree on the goal Self-interested agents may have private goals that can be in conflict with the plan goal but I do not deal with conflicting goals here (see [90] for an account on goal generation and planning in argumentation frameworks).
2. **Proposing stage**: This stage is where an agent takes the *proponent* role and puts forward the plan proposal to reach the goal. Whenever a plan is presented each proposal is evaluated separately. The protocol enables participants to present several proposals until one is accepted. The protocol assumes the selection of

Table 4.1: Comparison of Protocol Stages

Hulstijn [89] Negotiation	McBurney et al. [106] Deliberation	*PDGP* Persuasion and Deliberation
Opening	Open	Opening
Information	Inform	-
Proposing	Propose	Proposal
-	Consider	Evaluation
-	Revise	Refinement
-	Recommend	-
-	-	Change stage
Confirmation	Confirm	Selection
Closure	Close	Closing

the plan that agents put forward is an internal mechanism based on the agent's preferences. The protocol puts no constraints on how the agents behave in this sense. So, if an agent has two possible plans from which he can choose and both plans promote the same values the agent has to apply an internal process to select from these two [2].

3. **Evaluation stage**: This stage is where agents evaluates the plans using critical questions. For each proposal, agents acting as *respondents* can engage in a dialogue comprising questions and/or attacks over elements presented with the argumentation scheme for plan proposals *ASP* (see Section 3.4) to identify points that could make the argument scheme inapplicable.

4. **Change stage**: This is an intermediate stage to change the type of dialogue. Figure 4.3 shows which locutions change the dialogue stage from an *Evaluation* stage (questioning and challenging) to a *Refining* stage (deliberation about specific actions) and back again. This stage is the entry point for a deliberation process where agents can engage in a dialogue about what to do once a proposed action

[2] In the implementation presented in Chapter 6 agents put forward their preferred plans and if two promote the same values an arbitrary plan is selected.

from the *Evaluation* stage is found to be not valid not possible or can have an alternative.

5. **Refinement stage**: If a plan proposal needs to be *refined* specific locutions enable this task. A *refinement* in the context of the protocol, means a change on the action level either removing, replacing or adding actions to the plan. The details on how agents selects or discard actions out of the scope of the protocol since this is a task performed by the planner. The locutions in this stage then allow agents to propose, discard or agree on actions in a plan based on replanning tasks. Refinement processes can be triggered by actions that are not possible or not valid that need to be replaced (see the critical questions about the validity of actions and actions' elements in Layers 1 and 2 of Section 3.5) or if an agent identifies an action or actions that can be added or replaced in the plan that benefit the proponent in some way. Once the plan is changed it can be evaluated using the plan checker.

   The *Evaluation* and *Refinement* stages constitute the core of the dialogue and are can to be entered as many times as necessary. The diagram in the next section exemplifies their use.

6. **Selection stage**: After the agents have accepted on whether a plan is in principle unquestionable, they have the possibility to agree or not. The dialogue has two possible outcomes: either a consensus for a plan execution is reached or not. In the case when equally good plans promote the same value at the end of the dialogue, the process of selection depends on the proponent agent which selects the plan which promotes more values, or when these are equal, the plan that demotes fewer values. The same mechanism is applied when equally good plans are preferred by distinct agents. If this rule does not output a plan an arbitrary plan is selected by the protocol. The aim of the protocol is to allow agents to discuss several proposals and evaluate them through dialogue. This mechanism to agree on a plan can be extended allowing agents to present multiple proposals in the *Selection* stage and create an argumentation framework to select a plan according to an audience (cf. Section 2.2.2.2), this feature is not considered in this protocol.

7. **Closing stage** : At this stage agents finish their participation in the dialogue following an acceptance or rejection of a proposal.

Figure 4.1 presents the protocol stages as a sequence. The *Evaluation* and *Refinement* stages are where the proposals are challenged and possibly modified. Dialogue stages present several advantages in terms of the protocol specification and the implementation of it, for example, each stage reduces the locutions allowed for the agents depending on its role.

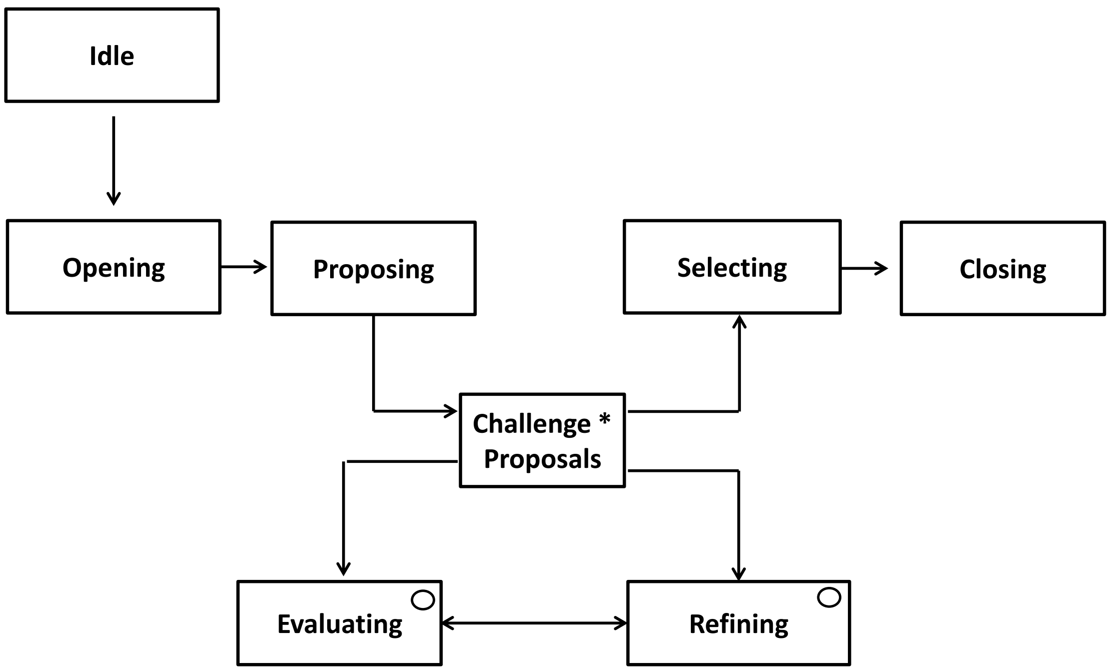


Figure 4.1: *PDGP* as a sequence of states. Boxes represent the dialogue stages, '*' indicates iteration and 'o' indicates selection.

4.1.2 *PDGP* Rules

In [111], McBurney and Parsons defined the following rules to specify a Dialogue Game specification: rules for the commencement of the dialogue, rules for the locutions and their combination, rules for commitments, rules for combination of commitments, rules for the speaker order and rules for the termination of the dialogue (these elements are presented fully in Section 2.2.3 of the Literature Review). Based on these types of rules the rules of the *PDGP* protocol are:

1. **Rules for the commencement of the dialogue**.
   The dialogue starts when an authorized agent creates the dialogue thread. The authorization to engage in the dialogue should be given by the social context. It is assumed that this verification is made before agents engage in a dialogue within the protocol.

2. **Rules for the locutions and their combination**.
   The dialogue stages constrain the locutions at each stage (see Figure 4.2). Agents cannot repeat moves with the same information e.g. one agent repeatedly questioning the same element of a plan. Also, a proponent agent is not allowed to

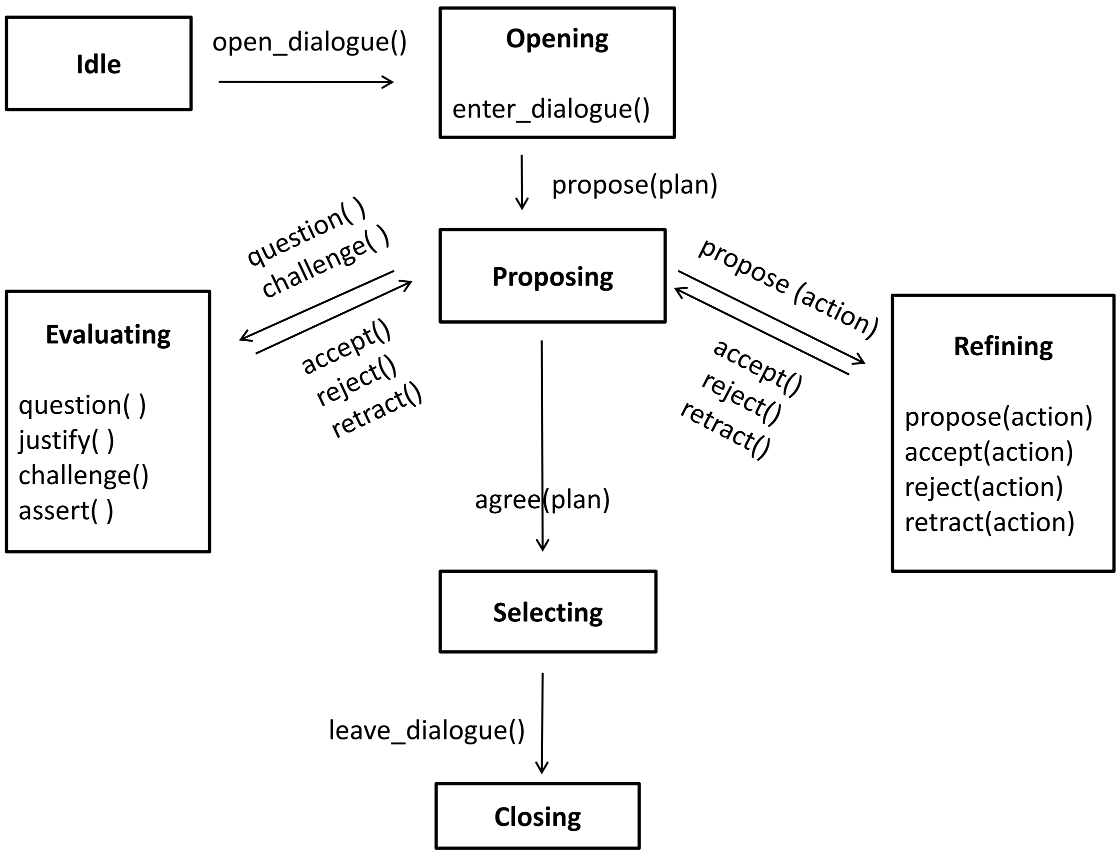


FIGURE 4.2: *PDGP* as a state transition diagram. Boxes represent the dialogue stages, arrows represent speech acts (the *Change* stage is omitted).

propose the same plan twice and a respondent agent is not allowed to question the same element twice.

3. **Rules for commitments**

   - Agents have a public commitment stores for each dialogue.
   - An agent is committed to an element if the agent asserts the element in the dialogue through a proposal or an assertion and the agent can drop commitments retracting its assertions (further details are given in the next section).
   - when an agent presents a proposal a *dialectical commitment* is created for each element of the proposal. I use here the term *dialectical commitment* to refer to commitments inside the dialogue i.e. the obligation of an agent participant of the the dialogue to defend an assertion when attacked by other participant. For example, if the *NGO* agent presents the current circumstances as $inZone(NGO, 1)$ it has the commitment to defend the belief it is attacked.

4. **Rules for the speaker order**
   The protocol stages, the agent roles and the locution post-conditions restricts to

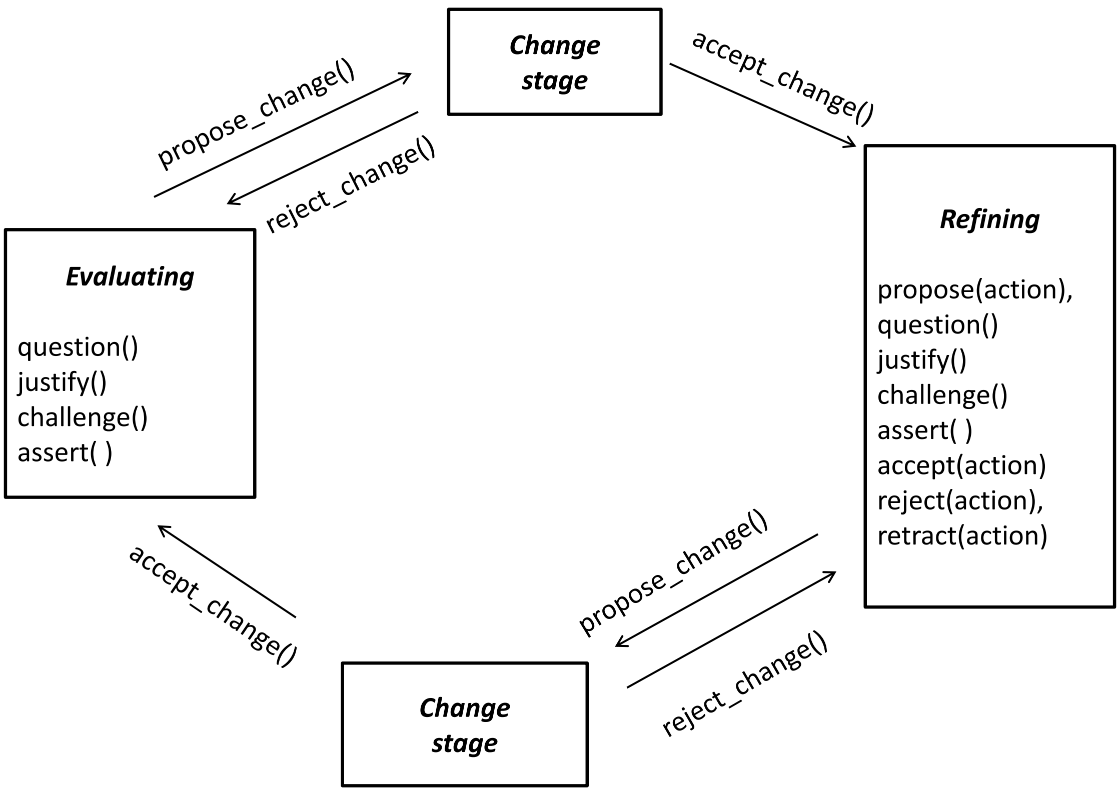


FIGURE 4.3: Change of stage locution flow to refine the plan in *PDGP*.

some extent the next speaker but do not explicitly indicate the next speaker in the dialogue. In the dialogue simulations implemented in Chapter 6 each agent speaks and waits for the other agent to issue a locution.

5. **Rules for the termination of the dialogue**
   The dialogue finishes when:
   - A plan is agreed on by all the participants.
   - When an agent withdraws from the dialogue prematurely and only leaves one agent in the dialogue.
   - When no more proposals are available and no accepted plan has been agreed on.
   - When the agents have accepted one or two proposals, then they should agree to one of the accepted plans based on their preferences.
   - When one agent decides not present a plan the protocol moves to the *Selection* stage and the dialogue ends accordingly (the protocol gives the participants the opportunity to present at least one plan each).

4.1.3 *PDGP* Syntax

The protocol uses the *Fatio* protocol locutions [110] (discussed in Section 2.2.3.5) which extends the *FIPA-ACL* [70] locutions to handle rational argumentative dialogues.

The language syntax presented comprises two layers as in the *FIPA-ACL* specification [70]. The outer (wrapper) layer comprises the locutions which express the illocutionary force of the inner content (the speech acts) and the inner layer relates to the topic of the discussion. The syntax of the speech acts of the *PDGP* protocol is presented in the following list with an example related to the example from Appendix A:

L1: *open_dialogue* (Ag, D_n)

The action of opening a dialogue D by agent Ag to create the dialogue thread. Example:

(*open_dialogue*(
    :sender: (agent-identifier: NGO)
    :content(
    dialogue: (dialogue-identifier: D_1)
)

L2: *enter_dialogue* (Ag, D_n)

The action of joining a dialogue D by agent Ag. As in the previous locution, the assumption is that a constraint in the *social context* validates that the agent has the permission to join the dialogue. Example:

(*enter_dialogue*(
    :sender: (agent-identifier: PKF)
    :content(
    dialogue: (dialogue-identifier: D_1)
))

L3: *propose*($Ag_{PRO}, D_n, Prop_n$)

The action of submitting a plan proposal $Prop_n$ by agent Ag_{PRO}. The proposal being of the form of the argumentation scheme ASP in Section 3.4 containing: a set of constraints applicable to the proposal, the current circumstances, the plan as a set of temporal actions for each agent, the new circumstances, the goal of the plan and the values related (the example plan presented is PL_1 in Table A.21):

```
(propose(
    :sender: (agent-identifier: $NGO$)
    :content(
    (dialogue: (dialogue-identifier:$D_1$))
    (proposal: (proposal-identifier: $Prop_{PL1}$))
   (
      (context(
        ($travelWarningZones(accompanied)$)))
      (current-circumstances(
        ($inZone(NGO, 1)$))
        ($inZone(PKF, 3)$)
        ($securedZone(1)$))
        ($securedZone(3)$))
        )
      (plan(
        (action($j_{mm}$ at $t_0$))
        (action($j_{im}$ at $t_3$))
        (action($j_{mm}$ at $t_6$))
        )
      (new-circumstances(
        ($inZone(NGO, 6)$))
        ($inZone(PKF, 6)$)
      (goal(
        ($inZone(NGO, 6)$))
        ($inZone(PKF, 6)$)
    )
    (values(
    $promotes(v_{NGOsec})$
    $promotes(v_{mov})$
    $promotes(v_{sec})$
    $promotes(v_{tt})$
))
```

L4: *question(*$Ag_{RES}, D_n, Prop_n, CQ_n, Element$)

The locution allows a respondent agent (Ag_{RES}) to question an element (a condition, an action, an effect, a plan, a value) of the proposal to make a weak attack (i.e. one which does not involve any commitments for the questioner) on a asserted element (*Element*).

The question CQ_n is one of the critical questions presented in Section 3.5 and *Element* is the identifier of the element questioned. Each question has a type (see Section 3.5.1) and this type determines the type of question attack on the element *Element*. The example presented below for question *CQA01* assumes agent questions the validity of the action *move*():

(question(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    question: (question-identifier: *CQA01*)
    question-element: ($move(6, 7)$ at t_0)
))

L5: *challenge*($Ag_{RES}, D_n, Prop_n, CQ_n, Element, typeEvidence, \phi$)

A respondent agent Ag_{RES} challenges an element (*Element*) of the proposal with question CQ_n. This locution represents an attack on an element of proposal $Prop_n$. The difference between the *challenge*() and the *question*() locutions is that *challenge* represents a direct attack on an element backed up by evidence ϕ using the same critical question and could be seen as an *undercut* (cf. Section 2.2.2) that directly oppose the support of the argument for the plan.

In Section 2.3.2.1 I discussed how the burden of proof determines who has the obligation provide evidence to claims. This evidence could be in the form of a proposition, a proof, a trust certificate etc. In the implementation of Chapter 6 is evidence is in the form of a token representing a trusted and assumed to be true element. For example, when a respondent agent poses the critical question *CQA1. Is the action valid?* with the locution *question* the respondent expects some proof on the validity of the action. When posed with this locution it could be rephrased as *"The action is not valid"* with a proof that sustains the attack and commits to the invalidity of the action. In the example below I assume *invalidRoute*$(10, 11)$ is a valid proposition accepted as evidence:

(challenge(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    question: (question-identifier: CQA02)
    question-element: (question-element: *move*$(10, 11)$ at t_0)
    evidence: (evidence: *invalidRoute*$(10, 11)$)
))

L6: *assert*($Ag_{PRO}, D_n, Prop_n, CQ_n, typeEvidence, \phi$)

A proponent agent Ag_{PRO} asserts new information ϕ of type $typeEvidence$ related to the proposal $Prop_n$ and in response to a *challenge*() locution with question CQ_n. This assertion can be seen as a *reinstatement* (cf. Section 2.2.2.1) that restores a previous condition of the element attacked with CQ_n.

Agents could assert beliefs about their world (in the form of norms, circumstances), values, plans and goals to answer to questions (in [110] for example, the assert locution allows to present beliefs, conditions, effects, and values). For example, if an agent is questioned with: *Is there an alternative way to achieve the goal?*, the agent can respond with the assert locution and the evidence takes the form of a plan as in the plan proposal ASP. All the assertions add the commitment to the agents public commitment store.

Note: In the implementation presented Chapter 6, agents present alternative plans for evaluation with another plan proposal ASP and not as an element of the *assert* locution (see step 8 in Section 6.4)

The following example assumes agent PKF provide evidence for route (10,11).
Example:

(assert(
:sender: (agent-identifier: NGO)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    question: (question-identifier: CQA02)
    typeEvidence: (typeEvidence: *circumstance*)
    evidence: ($validRoute(10, 11)$)
))

L7: *justify*($Ag_{PRO}, D_n, Prop_n, CQ_n, Element, typeEvidence, \phi$)

A proponent agent Ag_{PRO} justifies with evidence ϕ an element of the proposal identified *questioned* with CQ_n. note the the protocol uses justify to *respond*() to a *question*() and *assert*() to respond to a *challenge*(). The following example assumes a valid certificate confirms the validity of action *move*(). In Chapter 6 instead of using the concept of *certificate* I use a the concept of a *token*. Example:

(justify(
:sender: (agent-identifier: NGO)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    question: (question-identifier: *CQA01*)
    question-element: ($move(6, 7)$ at t_0)
    typeEvidence: (typeEvidence: $norm$)
    evidence: (evidence: $validCertificate(move)$)
))

L8: *accept*($Ag_{RES}, D_n, Prop_n$)

The action of accepting a proposal by respondent agent Ag_{RES}. Example:

(accept(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
))

L9: *reject*($Ag_{RES}, D_n, Prop_n$)

The action of rejecting a proposal $Prop_n$ by a respondent agent Ag_{RES} based on a successful attack. Example:

```
(reject(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D1)
    proposal: (proposal-identifier: Prop1)
))
```

L10: *retract*($Ag_{PRO}, D_n, Prop_n$)

A proponent or respondent agent Ag_{PRO} retracts a previous proposal allowing a previous dialectical commitment to be retracted. Example:

```
(retract(
:sender: (agent-identifier: NGO)
content(
    dialogue: (dialogue-identifier:D1)
    proposal: (proposal-identifier: Prop1)
))
```

L11: *propose_change*($Ag, D_n, Prop_n, D_s$)

A proponent or respondent agent Ag proposes a change of dialogue type to D_s to either deliberate about an action or return to the proposal stage (the agents' roles could change when entering the refinement stage). Example:

```
(propose-change(
:sender: (agent-identifier: NGO)
content(
    dialogue: (dialogue-identifier:D1)
    proposal: (proposal-identifier: Prop1)
    stage: (dialogue-stage: Refining)
))
```

L12: *accept_change*($Ag, D_n, Prop_n, D_s$):

The action of accepting a change of dialogue stage to D_s by agent Ag. Example:

```
(accept-change(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D1)
    proposal: (proposal-identifier: Prop1)
    stage: (dialogue-stage: Refining )
))
```

L13: *reject_change*$(Ag, D_n, Prop_n, D_s)$

A proponent or respondent agent Ag rejects the change of dialogue stage to D_s. Example:

(reject-change(
:sender: (agent-identifier: PKF)
content(
  dialogue: (dialogue-identifier:D_1)
  proposal: (proposal-identifier: $Prop_1$)
  stage: (dialogue-stage: *Refining*)
))

L14: *propose*$(Ag_{PRO}, D_n, Prop_n, Ac_i)$

The action of adding to the proposal $Prop_n$ an action Ac_i by the proponent agent Ag_{PRO}. I assume the information about the action scheduling is part of the action, i.e. how the action fit into the modified plan. This information then does not need to be part of the locution's syntax. Example:

(propose(
:sender: (agent-identifier: NGO)
content(
  dialogue: (dialogue-identifier:D_1)
  proposal: (proposal-identifier: $Prop_1$)
  action: (action(j_im))
))

where j_{im} is ($idle(NGO, 7)$ at t_5, $move(PKF, 7, 10)$ at t_5)

L15: *accept*($Ag_{RES}, D_n, Prop_n, Ac_i$)

The action of accepting action Ac_i as part of proposal $Prop_n$ by the respondent Agent Ag_{RES}. Example:

(accept(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    action: (action-identifier($action(j_i m)$))
))

L16: *reject*($Ag_{RES}, D_n, Prop_n, Ac_i$)

The action of rejecting action Ac_i from proposal $Prop_n$ by a respondent agent Ag_{RES} following a successful attack. Example:

(reject(
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
    proposal: (proposal-identifier: $Prop_1$)
    action: (action-identifier: action($j_i m$))
))

L17: *retract*$(Ag_{PRO}, D_n, Prop_n, Ac_i)$

Agent Ag_{PRO} retracts a previously proposed action Ac_i. Example:

```
(retract(
:sender: (agent-identifier: $NGO$)
content(
    dialogue: (dialogue-identifier:$D_1$)
    proposal: (proposal-identifier: $Prop_1$)
    action: (action-identifier: action($j_i m$))
))
```

L18: *agree*$(Ag, D_n, Prop_n)$

The action of agent Ag agreeing on a proposal $Prop_n$. Example:

```
(agree(
:sender: (agent-identifier: $PKF$)
content(
    dialogue: (dialogue-identifier:$D_1$)
    proposal: (proposal-identifier: $Prop_1$)
))
```

L19: *leave_dialogue*(Ag, D_n)

The action of leaving the dialogue D by agent Ag. Example:

$(leave\_dialogue($
:sender: (agent-identifier: PKF)
content(
    dialogue: (dialogue-identifier:D_1)
))

4.1.4 *PDGP* Semantics

The semantics for the protocol defines the preconditions which must hold before the locution can be uttered and post-conditions which apply following its utterance. I now present axiomatic semantics that define the preconditions and post-conditions for each locution.

R1: $open\_dialogue(Ag, D_n)$

Preconditions

- Agent Ag authorized to open a dialogue.
- Dialogue stage: *Idle*
- It is assumed that a constraint in the *social context* validates the agent has the permission to open the dialogue.

Postconditions

- Dialogue D_n created.
- Dialogue stage: *Opening*
- Agent Ag role: *Participant*

R2: $enter\_dialogue(Ag, D_n)$

Preconditions

- Dialogue D_n created.
- Dialogue stage: *Opening*
- Ag authorized to enter the dialogue.

Postconditions

- Dialogue stage *Opening*
- Agent Ag role: *Participant*

R3: $propose(Ag_{PRO}, D_n, Prop_n)$

Preconditions

- Dialogue D_n created
- Dialogue stage: *Opening*
- Agent Ag_{PRO} role: *Participant*
- At least two participants in dialogue D_n

Postconditions

- Dialogue stage: *Proposing*
- Agent Ag_{PRO} role: *Proponent*
- Agent Ag_{PRO} committed to all the elements in $Prop_n$.
- $Prop_n$ asserted.

R4: $question(Ag_{RES}, D_n, Prop_n, CQ_n, Element)$

Preconditions

- Dialogue D_n created.
- Dialogue stage: *Proposing,Evaluating/Refining*
- Agent Ag_{RES} role: *Respondent/Participant*
- Proposal $Prop_n$ asserted

Postconditions

- Dialogue stage: *Evaluating/Refining*
- Agent Ag_{RES} role: *Respondent*

R5: $challenge(Ag_{RES}, D_n, Prop_n, CQ_n, Element, typeEvidence, \phi)$

Preconditions

- Dialogue D_n created
- Agent Ag_{RES} role:*Respondent/Participant*
- Dialogue Stage: *Evaluating/Refining*
- Proposal $Prop_n$ asserted.

Postconditions

- Dialogue stage: *Evaluating/Refining*
- Agent Ag_{RES} role: *Respondent*
- Ag_{RES} committed to ϕ

R6: $assert(Ag_{PRO}, D_n, Prop_n, CQ_n, typeEvidence, \phi)$

Preconditions

- Dialogue D_n created
- Agent Ag_{PRO} role: *Proponent*
- Dialogue stage: *Evaluating*
- Proposal $Prop_n$ asserted.
- An agent Ag_{RES} issued question CQ_n.

Postconditions

- Dialogue stage: *Evaluating*
- Agent Ag_{PRO} committed to ϕ

R7: $justify(Ag_{PRO}, D_n, Prop_n, CQ_n, \phi)$

Preconditions

- Dialogue D_n created.
- Agent Ag role: *Proponent*
- Dialogue stage: *Evaluating/Refining*
- Proposal $Prop_n$ asserted.
- Ag_{RES} challenge with CQ_n.

Postconditions

- Agent Ag role: *Proponent*
- Dialogue Stage:*Evaluating/Refining*
- Ag_{PRO} committed to ϕ.

R8: *accept*$(Ag_{RES}, D_n, Prop_n)$

Preconditions

- Dialogue D_n created.
- Agent Ag role: *Respondent*
- Dialogue stage: *Proposing/Evaluating/Refining*
- Proposal $Prop_n$ asserted.
- Ag_{PRO} committed to elements in $Prop_n$.

Postconditions

- $Prop_n$ accepted.
- Dialogue stage: *Proposing*
- Ag_{RES} committed to elements in $Prop_n$.

R9: $reject(Ag_{RES}, D_n, Prop_n, \phi)$

Preconditions

- Dialogue D_n created.
- Agent Ag_{RES} in dialogue D_n.
- Agent Ag_{RES} role: *Respondent*
- Dialogue stage: *Proposing/Evaluating*
- Proposal $Prop_n$ asserted. Ag_{PRO} committed to $Prop_n$.

Postconditions

- Agent Ag_{RES} role: *Participant*
- Dialogue stage: *Proposing*
- Proposal $Prop_n$ rejected.
- Ag_{RES} commitments related to $Prop_n$ dropped and committed to ϕ.

R10: $retract(Ag_{PRO}, D_n, Prop_n)$

Preconditions

- Dialogue D_n created.
- Agent Ag_{PRO} in dialogue D_n.
- Proposal $Prop_n$ asserted.
- Agent Ag_{PRO} role: *Proponent*
- Dialogue stage : *Proposing/Evaluating*
- Agent Ag_{PRO} committed to $Prop_n$.

Postconditions

- Agent Ag_{PRO} role: *Participant*
- Dialogue stage: *Proposing*
- Agent Ag_{PRO} commitments related to $Prop_n$ dropped.

R11: $propose\_change(Ag_{PRO}, D_n, D_{stage})$

Preconditions

– Dialogue D_n created.

– Agent Ag_{PRO} in dialogue D_n.

– Dialogue stage: *Evaluating/Refining*

– Agent Ag_{PRO} role: *Proponent/ Respondent*

Postconditions

– Agent Ag_{PRO} role: *Proponent/ Respondent*

– Dialogue stage: *Changing stage*

– Agent Ag_{PRO} role: *Proponent*

R12: $accept\_change(Ag_{RES}, D_n, D_{stage})$

Preconditions

– Dialogue D created.

– Agent Ag_{RES} in dialogue D_n.

– Dialogue stage: *Changing stage*

– Agent Ag role: *Proponent/Respondent*

– $propose\_change()$ asserted by Ag_{PRO}

Postconditions

– Dialogue stage: D_{stage}

– Agent Ag_{RES} role: *Respondent*

– Dialogue stage: Evaluating/Refining

R13: $reject\_change(Ag_{RES}, D_n, D_{stage})$

Preconditions

- Dialogue D_n created
- Dialogue stage: Evaluating/Refining
- Agent Ag_{RES} in dialogue D_n
- $propose - change()$ asserted by Ag_{PRO}

Postconditions

- Dialogue stage: Evaluating/Refining

R14: $propose(Ag_{PRO}, D_n, Prop_n, Ac_i)$

Preconditions

- Dialogue D_n created
- Agent Ag_{PRO} in dialogue D_n
- Dialogue stage: *Refinement*
- Proposal $Prop_n$ asserted

Postconditions

- Dialogue stage: *Refinement*
- Agent Ag_{PRO} committed to action Ac_i

R15: $accept(Ag_{RES}, D_n, Prop_n, Ac_i)$

Preconditions

- Dialogue D_n created.
- Agent Ag_{RES} in dialogue D_n
- Dialogue stage: *Refining*
- Proposal $Prop_n$ asserted
- Ag_{PRO} committed to action Ac_i

Postconditions

- Dialogue stage: *Refining*
- Ag_{RES} committed to action Ac_i

R16: $reject(Ag_{RES}, D_n, Prop_n, Ac_i, \phi)$

Preconditions

- Dialogue D_n created .
- Agent Ag_{RES} in dialogue D_n
- Dialogue stage: *Refining*
- Proposal $Prop_n$ asserted
- Ag_{PRO} committed to Ac_i

Postconditions

- Dialogue stage: *Refining*
- Ag_{RES} committed to element ϕ

R:17 $retract(Ag_{PRO}, D_n, Prop_n, Ac_i)$

Preconditions

– Dialogue D_n created

– Agent Ag_{PRO} in dialogue D_n

– Proposal $Prop_n$ asserted

– Agent Ag_{PRO} role: *Respondent*

– Dialogue stage: "Refining"

– Agent Ag_{PRO} committed to action Ac_i

Postconditions

– Agent Ag_{PRO} role: *Participant*

– Dialogue stage: *Refining*

– Agent Ag_{PRO} not committed to action Ac_i

R18: $agree(Ag_{RES}, D_n, Prop_n)$

Preconditions

– Dialogue D_n created

– Agent Ag_{RES} in dialogue D_n

– Agent Ag_{RES} role: *Respondent*

– Dialogue stage: *Proposing*

– Proposal $Prop_n$ asserted

Postconditions

– Agent Ag_{RES} role: *Participant*

– Dialogue stage: *Selecting*

– Agent Ag_{RES} committed to all the elements in the proposal $Prop_n$

R19: $leave\_dialogue(Ag, D_n)$

Preconditions

– Dialogue D_n created

– Agent Ag in dialogue D_n

Postconditions

– Ag is not participant of dialogue D_n

– Ag commitments removed from the commitment store

4.2 Protocol Evaluation

In [112], McBurney et al. defined a number of desiderata for argumentation protocols based on research on agent interactions, auction mechanism assessments, and elements from argumentation theory. In this section a discussion on the PDGP protocol characteristics is presented based on the desired features in [112].

- Stated dialogue purpose: The agents engaged in this dialogue are doing so to agree on a plan, so the purpose of *PDGP* is to agree on a multi-agent plan.
- Diversity of individual purposes: Agents are able to pursue their own goals in so far as they do not affect the common goal and the protocol does not prevent this.
- Inclusiveness: The rules to include agents in the dialogue are assumed to be outside of the protocol rules. These are given possibly by a set of norms applicable to the context in which the dialogue develops.
- Transparency: Protocol syntax and semantics are public and available to all the participants involved.
- Clarity of argumentation theory: The protocol conforms to the argumentation theory of Walton [186] where proposals are asserted through an argumentation scheme (see Section 3.4) and can be challenged by a set of related critical questions (see Section 3.5) through a dialogue game based on persuasion and deliberation locutions. The arguments presented through the protocol are not used to create an argumentation framework. This feature could be indeed a characteristic of the protocol but I am not considering this feature here. The protocol puts no constraints on how the agents behave in relation to the evaluation of the arguments proposed.
- Separation of syntax and semantics: Syntax and semantics are defined separately. The semantics is public and verifiable.
- Encouragement of resolution: The protocol enables agents to reach a resolution and all the termination paths lead to either the acceptance or rejection of the proposal. Some special cases are considered for example, when an agent withdraws from the dialogue prematurely and only leaves one agent in the dialogue (see Section 4.1.2, rules for the termination of the dialogue). The protocol encourages resolution in the sense that: (1) there exist the necessary locutions to reach an agreement given that there is at least one possible plan available, and (2) the protocol forces to select a plan in cases where preferred plans are not agreed upon.
- Discouragement of disruption: The protocol allows agents to leave the dialogue at any time but there is no disruption if sufficient agents remain in the dialogue. When an agent leaves the dialogue the remaining agents could continue to evaluate or refine a proposal. The dialogue could be disrupted if all the agents leave

the dialogue before agreeing on a plan. If the *Proponent* agent leaves his commitments are dropped and the thread for the related proposal ends giving the chance to the remaining agents to propose new plans. If one agent has all the vital information (about possible routes, for instance) and this agent leaves the dialogue the dialogues would revolve around the plans presented, if none of the remaining agents can provide an acceptable plan the dialogue would finish with no outcome. The protocol does not prevent autonomous agents from leaving the dialogue, only validates the locutions agents put forward.

- Self-transformation: The protocol allows assertions to be retracted through specific locutions enabling self-transformations, and so enables an agent to express revisions to its beliefs or intentions.

- System simplicity: The protocol has *eighteen* locutions which may not seem simple to handle but having dialogue stages makes it simpler to issue locutions at a particular stage. The protocol is intended for machine execution and this is relatively a small number of locutions, e.g. it is less than *FIPA-ACL*, and less than the legal responses in HTML. Also with the use of stages the locutions available are constrained.

4.3 Dialogue Examples

In this section I present two dialogue examples based on the the *NGO* Force scenario from Appendix A using the *PDGP* protocol. The dialogues are presented with a natural language description of the locution followed the *PDGP* notation. In Tables 4.2 and 4.3 agents engage in a dialogue about plans PL_A and PL_B (Tables A.21 and A.17 respectively) to reach zone 6.

TABLE 4.2: Example Dialogue to reach zone 6

Dialogue Stage	Description
Opening	NGO: Open dialogue to reach zone 6 $\boldsymbol{open\_dialogue(NGO, D_1)}$ Role NGO: *Participant*
	PKF: Enter dialogue to reach zone 6 $\boldsymbol{enter\_dialogue(PKF, D_1)}$ Role NGO: *Participant*
Proposing	PKF: I propose PL_A (see Table A.21) to reach zone 6. $\boldsymbol{propose(PKF, D_1, Prop_{(PL_A)})}$ Role PKF: *Proponent*
Evaluating	NGO: Does performing plan PL_A have a side effect which demotes the value v_{tt}? $\boldsymbol{question(NGO, D_1, Prop_{(PL_A)}, CQSE01, v_{tt})}$ Role NGO: *Respondent*
	PKF: The plan does not demote value v_{tt}, the plan duration is 12 which is below the limit of 17 (see Table A.16) $\boldsymbol{justify(PKF, D_1, Prop_{(PL_A)}, v_{tt}, norm, planDuration(12))}$

Table 4.3: Example Dialogue to reach zone 6 (continuation)

Dialogue stage	Description
Proposing	NGO: I cannot travel without escort on route(2,6) (route(2,6) is a warning route, see Table A.2) therefore, action $move(NGO, 2, 6)$ at time t_7 is not possible (note that this condition is not known to agent PKF). $\boldsymbol{challenge(NGO, D_1, Prop_{(PL_A)}, CQA3, move(NGO2, 6), precondition, safeRoute(2, 6)})$ PKF: OK, I retract PL_A. $\boldsymbol{retract(PKF, D_1, PL_A)}$ Role PKF: *Participant* Role NGO: *Participant* NGO: I propose plan PL_B $\boldsymbol{propose(NGO, D_1, PL_B)}$ Role NGO: *Proponent* PKF: OK, I accept plan PL_B. $\boldsymbol{accept(PKF, D_1, PL_B)}$ Role NGO: *Participant*
Selecting	NGO: I agree on plan PL_B. $\boldsymbol{agree(NGO, D_1, PL_B)}$ PKF: I agree on plan PL_B (note that agent PKF does not found any inconsistencies in the plan, a method to identify relevant questions will be presented in the next chapter). $\boldsymbol{agree(PKF, D_1, PL_B)}$
Closing	PKF: Leave dialogue. $\boldsymbol{leave\_dialogue(PKF, D_1)}$ NGO: Leave dialogue. $\boldsymbol{leave\_dialogue(NGO, D_1)}$

In Tables 4.4 and 4.5, I present a dialogue that *modifies* the plan PL_G (Tables A.20 and A.24) omitting the locutions to enter and leave the dialogue. In plan PL_G, NGO travels under escort of PKF from zone 6 to zone 14 via zones 9,11, 12 and 13. While waiting in zone 13 for permission to enter zone 14 NGO prepares resources. Then NGO administers aid in zone 14 while PKF is idle. In plan PL_{G1}, NGO travels under escort of PKF from zone 6 to zone 12 via zones 9 and 11. PKF waits in zone 12 while NGO prepares resources in zone 12 and then travels to zone 14 via zone 13 and provides help in zone 14.

TABLE 4.4: Example Dialogue to reach zone 14

Dialogue stage	**Description**
Proposing	PKF: I propose plan PL_G (see Table A.24) to reach zone 14. $\boldsymbol{propose(PKF, D_2, Prop_{(PL_G)})}$ Role PKF: *Proponent*
Evaluating	NGO: Does plan demotes value v_{sec}? $\boldsymbol{challenge(NGO, D_2, Prop_{(PL_G)}, CQSE01, v_{sec}, circumstance, warningZone(9))}$ Role PKF: *Respondent* PKF: Zone 9 is not a warning zone thus the value v_{sec} is promoted. $\boldsymbol{assert(NGO, D_2, Prop_{(PL_G)}, CQSE01, circumstance, \neg warningZone(9))}$ At this stage agent NGO can propose a plan refinement to travel alone from route(12,13), a fact not know by agent PKF. This refinement is not motivated by a problem with some action but as a way to finish sooner the plan. NGO: OK, I have plan refinement $\boldsymbol{propose\_change(NGO, D_2, Prop_{(PL_G)}, D_{Refine})}$
Refining	PKF: OK, I accept the change which are the actions? $\boldsymbol{accept\_change(PKF, D_2, Prop_{(PL_G)}, D_{Refine})}$ NGO: I propose action $prepareResources(NGO, 12)$ and $idle(PKF, 12)$ at time t_{12}. $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, prepareResources(NGO, 12))}$ $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, idle(PKF, 12))}$ NGO: OK, I accept action. $\boldsymbol{accept(PKF, D_2, Prop_{(PL_G)}, prepareResources(NGO, 12))}$ $\boldsymbol{accept(PKF, D_2, Prop_{(PL_G)}, idle(PKF, 12))}$
Refining	NGO: I propose action $move(NGO, 12, 13), idle(PKF, 12)$ at time t_{13}. $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, move(NGO, 12, 13))}$ $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, idle(PKF, 12))}$ PKF: You need to be escorted on route (12,13). $\boldsymbol{challenge(PKF, D_2, Prop_{(PL_G)}, CQA03, move(NGO, 12, 13),}$ precondition,$\boldsymbol{warningRoute(12, 13))}$

Table 4.5: Example Dialogue to reach zone 14 (continuation)

Dialogue stage	**Description**
	NGO: Not for this route (12,13). I can travel the route without escort. $\boldsymbol{assert(NGO, D_2, Prop_{(PL_G)}, CQA03, move(NGO, 12, 13), precondition, safe(13, 14))}$
	PKF: OK I accept the action $\boldsymbol{accept(PKF, D_2, Prop_{(PL_G)}, move(NGO, 12, 13))}$
	NGO: I propose action $move(NGO, 13, 14), idle(PKF, 13)$ at time t_{13}. $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, move(NGO, 13, 14))}$ $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, idle(PKF, 12))}$
	PKF: You need to be escorted on route (13,14), the action is not possible $\boldsymbol{challenge(PKF, D_2, Prop_{(PL_G)}, CQA03, move(NGO, 13, 14), precondition, warningRoute(13, 14))}$
	NGO: Not for this route (13,14). I can travel the route without escort. $\boldsymbol{assert(NGO, D_2, Prop_{(PL_G)}, CQA03, move(NGO, 13, 14), precondition, safeRoute(13, 14))}$
	PKF: OK accept action $\boldsymbol{accept(PKF, D_2, Prop_{(PL_G)}, j_{mi})}$
	NGO: I propose action j_{hi} (help,idle) at time t_{19} $\boldsymbol{propose(NGO, D_2, Prop_{(PL_G)}, j_{hi})}$
	NGO: OK accept action $\boldsymbol{accept(PKF, D_2, Prop_{(PL_G)}, j_{hi})}$
Proposing	NGO: I accept plan PL_{G1} $\boldsymbol{accept(NGO, D_2, Prop_{(PL_{G1})})}$

4.4 Chapter Summary

The *PDGP* dialogue protocol is based on an extension of existing protocols used for persuasion in [10, 24] and deliberation in [106] that allows both persuasion and deliberation so that agents can agree on a plan. Agents critiquing plans should be able to engage in several dialogue types to agree on the best plan to execute. The *PDGP* protocol combines locutions from persuasion and deliberation dialogues to enable agents to argue in the context of a plan evaluation.

This chapter presented the syntax and semantics of the *PDGP* dialogue game protocol that allows to *propose* plans based on the argumentation scheme for plan proposals of Section 3.4 and *evaluate* the proposals using the critical questions from Section 3.5. The semantics were given in terms of preconditions and postconditions for each locution. Furthermore I presented two example dialogues based on the example from Appendix A that exemplifies the use of locutions in the context agents selecting and modifying plans. Chapter 5 consider strategies to identify and select questions in dialogues and Chapter 6 present an implementation of the *PDGP* protocol using *tuple centres* and dialogue simulations to enable agents to critique each others plans.

Chapter 5

Strategies to Select Critical Questions

Strategies for cooperation in distributed problem solving have been studied extensively in the past decades e.g. [37, 96]. This chapter considers the problem of creating a strategy [1] or mechanism to select a critical question in the context of agents engaged in a dialogue to select a plan. Cooperative dialogues about plans are complex in the sense that agents use different types of dialogue and discuss plans at different levels (cf. Section 2.4.3). The potential length of dialogues about plans when using argumentation schemes and critical questions motivates the need to define a methodology to *identify* and *select* an appropriate critical question.

The aim of the strategy is to reduce the number of questions exchanged in the dialogue and in consequence: (i) reduce the communication overload, (ii) accept a valid proposal or dismiss an invalid proposal faster and (iii) lead to new information needed to reach an agreement. Consequently, this strategy can improve the clarity and scrutability of automated dialogues.

In open environments it is in principle desirable that agents that are engaged in a dialogue have the freedom to pose any question, but in some contexts the agents may be restricted by preconditions imposed by the domain, the social context or the dialogue protocol (cf. Section 2.2.3 and the conceptualisation of dialogue protocols and conversation policies). While these restrictions reduce the freedom of the agents, on the other hand there are benefits in terms of the efficiency and coherence of the dialogue.

With the strategy presented in this chapter, autonomous agents can: (i) identify relevant questions according to the information presented beliefs of the agents and (ii) then prioritize those questions. Thus, the strategy does not restrict but enhance the selected question to be posed.

The remainder of this chapter is structured as follows: Section 5.1 presents a general approach to the concept of *strategy* in dialogues. Section 5.2 presents the main idea behind the concept of strategy developed in this thesis and relates it to the concepts

[1] The term "strategy" is used in the ordinary English sense of the word. No reference to the technical game theory arrogation of the term is intended.

defined in Chapters 3 and 4. Section 5.2.1 discusses the first step in the strategy which focuses on the identification of relevant questions depending on the information presents and the beliefs of the agent. Section 5.2.2 presents a prioritisation mechanism to select a critical questions to pose. This section presents two priority orderings for the critical questions giving reasons for it. Section 5.4 presents an example on how the strategies can be used referring to the example from Appendix A. Finally, Section 5.5 concludes the chapter with a discussion on the benefits and characteristics of the strategies.

5.1 Strategies in Dialogues

In the argumentative dialogue literature there is no clear consensus on the notion of *strategy* in a dialogue. The area of strategic agent argumentation in dialogues is interesting because of its importance in building agents that can successfully compete in domains characterised by open environments [50].

An approach to the concept of strategy is given in [150] where Rahwan et al. propose a methodology for designing strategies for negotiating agents. The approach relies on the idea that to create a strategy agents need to analyse their environment. The methodology focuses more on the strategy to maximise the outcome of the negotiation for agents and list tactics that can be used to create strategies e.g. seek to change a counterpart belief (persuasion as a strategy), gain a better understanding of the counterpart, seek to discuss a particular issue, etc. This approach is focused on a more general level decomposing objectives and composing of capabilities at the agent levels in order to produce tactics that eventually form strategies.

A strategy in a dialogue can be defined as the plan of action that indicates a sensible and advantageous move (locution) in favour of the agent or in the interest of dialogue coherence. The strategy then can be based on information from the context, from information of the opponent(s), from information of previous interactions, or from an internal reasoning mechanism that dictates to the agent what is the best possible move. Strategies have been studied and defined in relation to dialogues with three main approaches to the concept of strategy presented below.

1. The first approach is based on the concept that a move should be selected based on an *agent's local beliefs* about the world and the agent's capabilities.
   In [129], for example the strategy is based on the principle of divulging as little information as possible to other dialogue participants. The agents are then able to win arguments that it would not be able to win if complete information were available to all participants, since information which might be useful to the opponents can be suppressed.

2. The second approach constructs a strategy based on *other agent's beliefs*, such as their preference model as in [28] or their mental model as in [50].

In [28], for example, autonomous agents deliberating on the course of action consider the relative advantages and disadvantages of the different options. Agents have their own preferred outcome which may change as new information is received from the other agents involved in the deliberation. In essence the approach advocates a dialogue strategy designed to ensure that agreement reached is acceptable to each agent resolving or sharing their differing preferences. Thus, agents construct a strategy which takes into account the differences in their preferences.

Another example of this approach is given in [50] where the concept of strategy is presented in a persuasion dialogue context. The authors regard a strategy as a player's approach to making decisions. The dialogue game presented allows the proponent to win by exploiting not just defects in the opponent's reasoning or inconsistencies in the opponent's knowledge base, but also the incompleteness of its knowledge through an argumentation framework as presented in [56]. This identification of defects and inconsistencies provides a strategic sophistication in multi-agent dialogue.

3. A third approach is related to create a strategy based on *past dialogues* as presented in [65]. In this paper the authors present a decision-making mechanism where models of other agents are refined through evidence from past dialogues where these models are used to guide future argumentation strategy. The approach uses decision-theoretic and machine learning techniques to retrieve evidence from past dialogues.

A more comprehensive approach to the concept of strategy in dialogues is given in [5] where approaches 1 and 2 are combined. Amgoud and Hameurlain define a strategy as a decision problem that consist in selecting the locution or type of act to utter at a given step of a dialogue, and selecting the content that will accompany the act. The selection of the *locution* is then determined by the strategic goals and strategic beliefs of the agent. The selection of the *content* is determined by functional goals (directed to the subject of the dialogue) and the basic beliefs (about the subject of the dialogue). The strategic goals are defined as the meta-level goals of the agent and the strategic beliefs are meta-level beliefs about the dialogue or other agents. So, the strategy could be conceptualised as a two-step process where an agent meets his overall goals (with the locutions) and the dialogue goals (with the content of the locutions).

The concept of *strategy* presented in this chapter is developed at the *agent level* where the agents apply a rational process to select the content of a question to pose based on the local beliefs of the agent given that the communicative act, in this case $question()$ or $challenge()$, has been selected but the content remains to be determined.

5.2 Strategy to Select Critical Questions in Dialogues about Plans

The strategy is developed in the context agents engaged in a dialogue to agree on a multi-agent plan based on the argumentation scheme for plan proposal from Chapter 3 and the dialogue protocol from Chapter 4. The principal aim of the strategy is to dismiss an invalid proposal faster leading the participants to new information needed to reach agreement on a plan to execute. The strategy presented in this chapter works at the agent level where agents apply a rational process to select the content of a question to pose.

The strategy then focuses on the process of identifying relevant questions from the large set of critical question (cf. to Section 3.5) and selecting one to pose in the dialogue. The selected question then needs to be a valid question to pose (conforming to the protocol rules) and *appropriate* in the context of the dialogue. The mechanism is applied in the *Evaluating* and the *Refining* stages of the *PDGP* protocol .

A high level description on the way the strategy works when agents are presented with a plan proposal is the following:

- The respondent agent retrieves the available questions from the protocol
- The agent identifies the potential conflicts given the information presented
- The agent matches the conflict with a question to generate a list of questions to pose
- The agent add further questions not related to any conflict
- The agent prioritise the list of questions according to some criteria and select a question to pose

In consequence , the aim of the *strategy* is to reduce the communication overload by decreasing the number of questions needed to reach an agreement on a proposal or dismiss it through a process of selection and prioritisation. By defining this mechanism, the strategy can provide a characterisation of cases when some types of questions are more *effective* to finish the dialogue faster. I use the term *effective* in the sense of questions being *appropriate* to dismiss the proposal .

The strategy proposed is comprised of two main steps:

1. **Step 1**: The respondent agent identify questions relevant to the dialogue by:

    - identifying a conflict (contradicting evidence) between the respondent's representation of the world and the information presented the proposal (the respondent agent then will *challenge* elements it has evidence to back them up) or

   - identifying elements that match the scheme and does not *challenge* the proposal (questions that refer to alternative options or seek for more information related to the proposal).

2. **Step 2**: The respondent agent prioritises the critical questions and poses the question with highest priority. This prioritisation refers to the group similar questions together taking into account the nature of the question. Thus, it can be said that the strategy selects a critical question to pose from a set of relevant questions. Using this idea, agents can pose and resolve critical questions in a descending order of priority to promote efficiency while maintaining focus and relevance.

I now discuss these two steps in more detail.

5.2.1 Identifying relevant questions

In dynamic environments, agents engaged in dialogue could have different representations of the world, therefore, there is the need to have a mechanism to check these inconsistencies before agreeing on or modifying a plan. The respondent agent identifies questions to present in the dialogue based on the information in the proposal. When finding these questions the respondent agent is verifying the plan proposal presented against its local beliefs. The process identifies questions comparing or validating the information in the proposal which is: the goal, the initial state, the action specification(s), the constraints in the social context and the values involved.

Each validation is based on a critical question from Section 3.5. For example, in question *CQA01: Is action α a valid action?*, a process checks if the suggested action matches the definition in the respondent's local beliefs. If a different action representation is detected, the corresponding question is added to a list of questions to pose. I use the term "validity" in relation to actions to specify that agents agree that the action is available (cf. to Section where I explain the concept of validity for the critical questions).

The ontology alignment problem in fact is a different problem from the problem of differing beliefs about representation of the world. There may be an element of both problems in a scenario like the one presented in the example from Appendix A. Although I provide a mechanism to some extent align the ontologies of the agents, I do not provide a comprehensive mechanism to do this. For example in [175], the ontology alignment problem is discussed for a similar planning scenario where a modelling technique is presented to align a shared ontology through the definition of a common information model.

The main tasks when aligning the ontology in [175] are the following:

- Resolve inconsistencies in the initial circumstances
- Provide valid operators for the actions.

- Resolve inconsistencies for equal operators: add or remove a condition or an effect for an action or agree on the action duration.
- Resolve different domain objects.

Specifically, the respondent agent needs to examine:

- The current circumstances: this is the initial state of the world for the proponent agent. If the respondent agent does not have the same beliefs about the state of the world the plan presented may not be applicable. Furthermore, if an agent
- The action specifications: this validation is extended to all the action elements (preconditions, invariant conditions, termination conditions, start effects, end effects, cf. Section 3.2). If the action result is the same, other differences should not matter, but for the purpose of my approach any difference in the action specification generates a potential attack on the proposal.
- The constraints in the social context: e.g. *Does the proponent have the permission to initiate the dialogue?*
- The plan goal: ideally the plan goal should be agreed before the plan evaluation, but in some cases the plan goal needs to be agreed during the dialogue. In many real domains, the process of evaluation may lead to changes in the plan goal.
- The values involved in the plan proposal: although the order of values is subjective, agreement on the values which will be promoted is relevant.

I use the algorithm in Table 5.1 to identify the critical questions comparing the information in the proposal with the respondent's beliefs. Further details on the implementation of this algorithm are given in the next chapter.

In a cooperative dialogue scenario like the one I am considering, agents must agree on their beliefs about the world. In a continuously changing environment this may be very difficult. When agents agree at some point on a set of circumstances, a change in the environment could invalidate several coordination agreements between agents. If an agent identifies a change in the view of the world through its actualization loop the agent can pose for example the same question about the preconditions for an action.

The process to form questions identifies and helps to resolve conflicts in the world representation of the agents after which the plan selection process can be continued. The process of identifying questions takes into account the fact that some questions depend on the outcomes of others. For example, when checking for the validity of an action element, if the action is not valid there is no point in considering the question of whether the action is currently possible. From the running example, consider the question: *Is the precondition* $inZone(NGO, 2)$*, valid?*. If the respondent agent believes $inZone()$ is not a defined (which is not) the agent can discard issue the question and discard questions about the possibility of the precondition: (*e.g. Is the precondition possible at time* t_0) without further consideration.

Table 5.1: Pseudo-code to identify questions regarding the beliefs of the agents.

```
      Step                                Comment

1.    IdentifyConflicts()                 Information taken from the plan proposal
2.        VerifyPlanGoal()                Differences in the plan goal
3.        If difference detected
4.            AddQuestions()              Creates a list with questions to pose
5.        VerifyInitialState()            Differences in the initial state
6.        If difference detected
7.            AddQuestions()
8.        For each Action in the plan
9.            VerifyActionSpecification() Differences in the action specifications
10.           If difference detected
11.               AddQuestions()
12.       Next Action
13.       VerifySocialContext()           Differences in the social constraints
14.       If difference detected
15.           AddQuestions()
16.       VerifyValues()                  Differences in the values
17.       If difference detected
18.           AddQuestions()
19.   End
```

5.2.2 Critical Question Prioritisation

In a planning scenario I am considering the critical *critical questions* refer to the domain, to the plan or actions that comprise it, or to the timing of the actions (cf. Section 3.5.1). So, when agents agree agree on a plan first they need to resolve inconsistencies in the beliefs about the domain (to discuss valid plans), agents might lose time discussing the suitability or effectiveness of plans if they do not believe the elements that comprise the plan are valid. The time in which the actions are executed then can be discussed at last. The typical planning process in fact follows the same order: given a valid domain and problem representation a planner outputs a plan and then a scheduling process can be applied to it.

In this step I present an order in which the questions identified in the previous step can be prioritised in the dialogue. Once all the relevant questions have been identified, I investigate which questions should be put forward first to make the dialogue more efficient. I now present a more detailed analysis of the critical questions to classify and order them taking into account this finer grained description. The reason to categorise the critical questions is to identify their intrinsic purpose in the dialogue, and then use this purpose to give the questions a priority according to the particular strategy being used. To provide a prioritisation to questions in a dialogue I focus on the way a proposal could be questioned depending on the nature of the critical question. A critical question can challenge:

- The **validity** of an element: Questions in this category challenge an element (the plan, an action, a condition, an effect) on its legality or permissibility according to the context in which the dialogue occurs.
- The **possibility** of an element: Possibility differs from validity in the sense that an action may be valid or legal, but not possible given the *current* circumstances.
- The **possibility in time** of an element: The execution of an action could be possible but not possible at the time specified.
- The **suitability** of an element: Questions in this category challenge if an element (the plan, an action, etc.) is appropriate or effective in relation to the context in which the dialogue occurs. These questions include side effects of the plan, or circumstances not evident to all the agents.
- Other possible **alternatives**: These questions present a challenge to a suitable, valid and possible plan with other valid alternatives.

These categories provides a natural order in which questions can be posed. First the aim is to establish that the plan presented is **suitable** or reasonable based on the values it promotes. Examples of suitability questions are the following:

- *CQSE-01. Does the plan have a side effect which demotes the value v_n?* If a plan has side effects not considered regarding a specific value it may not be considered desirable by a particular audience.
- *CQPP-07. Is the goal already achieved?* It may not reasonable to agree on a plan if the intended circumstances already hold.
- *CQSE-02. Does performing the plan have a side effect which demotes some other value v_n?*
- *CQPP-06. Is the value v_n promoted by the plan?*

Once the suitability of the plan has been agreed, agents can focus on the **validity** of the elements to resolve any conflicts about the beliefs about the problem situation that the agents have, for example: *CQA-01. Is the action valid?* or *CQA-04. Are the action start effects valid?*

Once agents agree on the validity of the elements, the **possibility** and **possibility in time** of the plan can be addressed so that any differences in beliefs about the current situation can be resolved. An element can be valid but not possible for the current state of the world when an agent has the action in its repertoire but the preconditions are not currently satisfied, necessitating another step in the plan to enable the action. When possibility questions are considered, a finer grain of detail may be addressed leading us to consider the actions of the plan, for example:

- *Is the plan PL possible?* The plan PL may be recognised by the agents as a legitimate plan (based on the validity of the actions that comprise the plan) but it may not be "possible" to execute it at a certain train station or at a certain time.
- *CQAC-01. Could sequential actions Ac and Ac_1 be performed concurrently at some point?* This addresses the efficiency of the proposed plan.
- *CQAT-01. Is the action possible with the specified duration?* An action might not be possible with the specified duration.

Once agreed on the elements of the plan as a whole the analysis can continue over specific questions and can follow the same order, first suitability questions, then validity questions and finally possibility questions.

Finally, the strategy considers **alternative options**, either for actions or plans as a direct attack on the critiqued plan. The plan proposed may be valid and possible for all the agents involved but still another plan may be a better option given their preferences. For example:

- *CQAO-01. Is there an alternative plan PL_x to promote the same value v_n?*
- *CQAO-03. Is there an alternative plan to realize the same goal ?*
- *CQAO-05. Is there another action that could be performed with the same result?*

Following this analysis a priority order to consider critical questions in a dialogue could be determined by strategy $s1$. Figure 5.1 presents the critical question priority for strategy $s1$.

The priority ordering $s1$ may not be necessarily the most appropriate for all the contexts in which planning dialogues develop. Another priority order may be more effective in other situations. Another reasonable order can be placing "validity" questions before the "suitability" questions. It can be for example that since the belief alignment is an important factor in plan creation or modification "validity" critical questions need to be considered first. But in some scenarios suitability questions may not be an issue for agents or may not be strong enough to defeat an argument.

With some extra information agents could tailor the questions hat need to be asked for example, based on previous experiences in the same context agents can focus on questions that were successful to defeat an argument. Another example is if agents share an identical preference model agents can may ignore questions related to the plan preference.

Therefore present another ordering of question types $s2$, in Table 5.2 in which questions about the validity of the elements are considered first. In the next chapter I present an implementation where agents use both these strategies in dialogue simulations with various scenarios. I compare the results with a scenario with no strategy does no (i.e. that go through the first stage of finding the relevant questions to ask, see details of

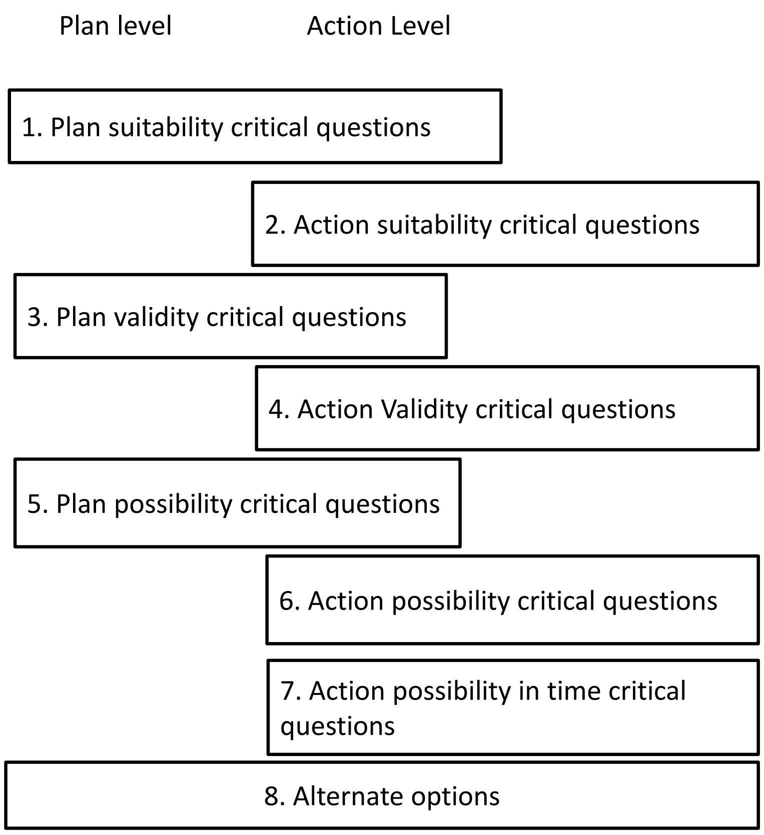

Plan level
Action Level
1. Plan suitability critical questions
2. Action suitability critical questions
3. Plan validity critical questions
4. Action Validity critical questions
5. Plan possibility critical questions
6. Action possibility critical questions
7. Action possibility in time critical questions
8. Alternate options

FIGURE 5.1: Strategy $s1$ critical question priority.

the implementation in Chapter 6) and with a random question selection and with a process to determine their comparative suitability, and to characterize when strategy $s1$ is preferred to strategy $s2$ and vice versa.

Table 5.2: Priority order in which questions are considered for both strategies.

	Strategy *s1*		**Strategy *s2***
1.	Plan suitability	1.	Plan elements' validity
2.	Actions' suitability	2.	Actions' validity
3.	Context suitability	3.	Context validity
4.	Plan elements' validity	4.	Plan suitability
5.	Actions' validity	5.	Actions' suitability
6.	Context validity	6.	Context suitability
7.	Actions' possibility	7.	Actions' possibility
8.	Plan elements' possibility	8.	Context possibility
9.	Alternative actions	9.	Alternative actions
10.	Alternative plans	10.	Alternative plans

5.3 Strategy preference

The strategies then can be preferred to deal with, for example:

- Inconsistencies in the agents' ontology (questions about the validity of elements).
- Inconsistencies in the action specification.
- Inconsistencies in values recognised by the agents.
- Domains where information changes fast (questions about the initial circumstances)
- Domains where the scheduling of actions could be a problem (questions about the scheduling of the actions).
- Scenarios where agents have different representations of the world (questions about the possibility of elements).
- Scenarios with reliable plans (alternate plans).

5.4 Example

In this section I present an example that details how the strategies in this chapter can be used using the example from Appendix A. Table 5.3 presents the conflicting beliefs of the agents related to elements in PKF's plan PL_E (see Table A.23) together with the question that is generated by agent NGO.

Table 5.4, presents all the questions identified by agent NGO with the algorithm from *Step 1* of the strategy (see Table 5.1). Different action representations and beliefs about propositions motivate agent justify for agent NGO to consider these questions

TABLE 5.3: Conflicting beliefs of agents related to elements in Plan PL_E.

	Element	**NGO**	**PKF**
1.	start effect of action $escort()$ at t_0 **generates CQA-05**	$GroundForce(deployed)$	$GroundForce(ready)$
2.	start effect of action $escort()$ at t_0 **generates CQA-04**	-	$flanksCovered()$
3.	end effect of action $escort()$ at t_0 **generates CQA-09**	$GroundForce(deployed)$	$GroundForce(vigilant)$
4.	precondition of action $moveEscorted()$ at t_4 **generates CQSE-01**	$secureZone(8)$	$inConflict(8)$
5.	precondition of action $moveEscorted()$ at t_4 **generates CQA-03**	$GroundForce(deployed)$	$GroundForce(vigilant)$

TABLE 5.4: Questions identified by agent NGO related to Plan PKF's plan PL_E.

(a) CQA-04	Is the start effect $flanksCovered$ of action $escort()$ valid? Question type: **validity**
(b) CQA-03	The precondition $GroundForce(vigilant)$ of action $moveEscorted(NGO, 6, 8)$ at t_0 is not possible Question type: **possibility**
(c) CQA-09	The end effect $GroundForce(ready)$ of action $escort(NGO, 6, 8)$ at t_0 is not possible. Question type: **possibility**
(d) CQA-03	The precondition $GroundForce(vigilant)$ of action $moveEscorted(NGO, 8, 14)$ at t_4 is not possible Question type: **possibility**
(e) CQA-03	Is the precondition $secureZone()$ of action $moveEscorted(NGO, 8, 14)$ at t_4 possible? Question type: **possibility**
(f) CQA-09	Is the end effect $GroundForce(ready)$ of action $escort(NGO, 8, 14)$ at t_4 possible? Question type: **possibility**
(g) CQSE-01	Are there any side effects from the execution of action $moveEscorted(NGO, 8, 14)$ at t_4? Question type: **suitability**
(h) CQAO-03	Is there an alternative plan to reach the same goal? Question type: **alternative options**

Table 5.5: Example priority order for questions in Table 5.4.

Strategy ordering *s1*	**Strategy ordering *s2***
Suitability questions (h) $CQA-03$	Validity questions (a) $CQA-04$
Validity questions (a) $CQA-04$	Plan suitability questions (h) CQA_03
Possibility questions (b) $CQA-03$ (c) $CQA-09$ (d) $CQA-03$ (e) $CQA-03$ (f) $CQA-09$	Possibility questions (b) $CQA-03$ (c) $CQA-09$ (d) $CQA-03$ (e) $CQA-03$ (f) $CQA-09$
Alternative options (h) $CQAO-03$	Alternative actions (h) $CQAO-03$

Table 5.5 presents the ordering of the questions according to the priority orderings in Table 5.2. In this case the ordering does not differ too much, just the first two questions change. The possible dialogue using the ordering from strategy $s1$ ends with just *two* questions: ((h)*CQA-03* and (a)*CQA-04*) since agent PKF do not have evidence to support the precondition $flankscovered$ covered by question (a)*CQA-04*). On the other hand the strategy ordering $s2$ finishes with just one question *(a) CQA-04* since this is asked first (as discussed in Section 3.1, agents rely on *evidence* to support the elements presented). While this dialogue example is very short, it describes how the *identification* and ordering of questions does matter in terms of the questions needed to end the dialogue. Without identifying the questions from *step 1* the agent would have to pose all the questions until finding the one the defeats the element presented. The *step 2* groups the questions and constructs a sequence in which questions can be posed. Chapter 6 describes an implementation of these strategies and how different orderings perform differently in terms of the number of questions needed to end a dialogue and thus promote efficiency.

5.5 Chapter Summary

Cooperative dialogues about plans are complex in the sense that agents use different types of dialogue and discuss plans at different levels. In this chapter I considered the problem of choosing a strategy for selecting critical questions in the context of a cooperative dialogue for agents discussing plans. I presented a mechanism to identify, prioritize and select questions in automated dialogues. The identification of relevant questions provides the necessary questions to align the beliefs on agents. So, the first step in the strategy is the identification of inconsistencies between the information in the proposal and the respondent agent's beliefs. Once the relevant questions have been identified, the mechanism prioritizes the questions for the respondent agent to pose a relevant one in the dialogue.

The advantages of the the strategy (question identification and prioritization) can be summarized as follows:

- The strategy identifies the elements that can potentially cause a conflict and maps them to critical questions.
- The strategy provides a priority order in which questions can be posed that helps to explain the reasoning process of the agents.
- The strategy separates clearly the process of identifying questions from the process of question prioritization.

The detail and length in dialogues about plans using argumentation schemes and critical questions motivates the need to use a strategy to select critical questions in order to reduce the number of questions exchanged and consequently improve the scrutability of dialogues. I presented a classification of questions that allow us to identify their nature. Based on this classification I presented two prioritisation orders that agents can use in automated dialogues. The approach to develop strategies is based on identifying which types of questions are the most relevant to bring benefits to the dialogue in terms of efficiency and plan selection. Critical questions are given a priority according to their type and taking into account the factors that could represent points of disagreement in specific scenarios. Although different prioritizations in the strategy may work better in different scenarios I believe the ordering priorities provided help to tease out the problems in the plan.

Some authors consider a strategic dialogue approach based on other agents' value preference models [28], identifying avoidable mistakes when expanding the opponents knowledge base [50], or taking evidence from past dialogues to form a strategy on the most persuasive argument [65] as mentioned earlier in this chapter. All these approaches are based on a model of other agents' beliefs, desires or intentions. Even though the strategy works from a different perspective and on a different level, I believe the two approaches could benefit each other. Embedding other agents' model of preferences in the prioritising function could allow the respondent agent to select an even more relevant question. Specifically, if an agent knows that the proponent agent preference for an action is low, the respondent agent would avoid presenting an alternate option for this action in the first instance. Any theory of how to model other agents in the dialogue could be used to augment either of the generic strategies proposed here. It could be argued that for automated machine dialogues that the number of questions is not relevant and so a strategy to select a question is not relevant. I believe the use of a strategy in terms of question relevance can improve the quality of the dialogue and avoid communicating redundant information between agents that can be a significant communication overhead.

In Chapter 6, I present an implementation of these strategies and in Chapter 7 analysis of the results of dialogue simulations in terms of the number of questions issued with each strategy. As shown in the results of the evaluation in Chapter 7, I argue that the use of a strategy when selecting a question in a dialogue regarding plans is beneficial, an thus different strategies perform better in different cases.

Chapter 6

Implementation

In this chapter I present the implementation of a system where two agents engage in a dialogue using the argumentation scheme from Chapter 3, the dialogue protocol from Chapter 4 and the strategies from Chapter 5. The implementation also shows with experiments that using a strategy to identify and prioritise questions agents exchange less questions and they reach agreements on a plan more quickly. The implementation uses a scenario where two agents need to agree on a plan to travel together.

In the test cases designed agents engage in simulated dialogue runs having for each run different plans, different beliefs about the world, different preferences (ordering of values) about plans and different strategies. In the dialogue simulations agents propose plans using the dialogue protocol from Chapter 4 to persuade each other and agree on a possible plan for both. Each proposal contains a plan that could be questioned and/or attacked according to the critical questions embedded in the protocol. The questions are used to accept or reject the proposal based on the validity, possibility and suitability of the elements presented. The strategies from Chapter 5 are also implemented and compared with a scenario where agents do not have a strategy and questions are posed randomly to provide a comparison point.

The remainder of this chapter is structured as follows: Section 6.1 describes the dialogue protocol implementation, Section 6.2 describes the agent and the system architecture, Section 6.3 describes the design of the test cases to and finally, Section 6.4 explains how the dialogue simulations were implemented.

6.1 *PDGP* Implementation

6.1.1 *TuCSoN* and *ReSpecT*

A blackboard system is an Artificial Intelligence application where a common knowledge base, the *blackboard*, is iteratively updated by a diverse group of specialist knowledge sources [85]. To implement the dialogue game protocol I used *TuCSoN* (Tuple Centres Spread over the Network), a blackboard software platform that enables communication and coordination of distributed/concurrent agents through *tuple centres* [127].

Tuple-based coordination models originate from the field of paralleling programming, but their features are useful for the coordination of distributed systems e.g. *tuple spaces* [72, 74, 203]. The *Tuple space* model was introduced originally as a powerful way of realising communication and coordination in concurrent systems, alternatively to message-based and shared-variable based models. *Tuple spaces* are also used to provide communication and coordination features for agents using a generic communication style.

The infrastructure of *TuCSoN* supports agent communication and coordination by providing *tuple centres*, which are shared and reactive information spaces, distributed over the infrastructure nodes. A tuple is considered in this context as an ordered collection of heterogeneous information data. *TuCSoN*is implemented in Java and provides a set of APIs (Application Programming Interface) to interact with *tuple centres.*

A *tuple centre* is a shared and reactive space distributed over the infrastructure where agents insert, retrieve and read information in the form of *tuples. Tuple centres* specialise and extend the basic *tuple space* model by using logic tuples. A *tuple centre*, in contrast to a *tuple space*, is enhanced with a behaviour specification that defines the responses (reactions) to communication events. A *tuple centre* is thus an enhancement of a *tuple space* where agents synchronise and cooperate over information available in a shared data space with a programmable reactive behaviour using first order logic terms [126].

Some of the advantages that *tuple centres* offer are the following:

- Spaces with a reactive behaviour are programmable by humans or agents.
- Information representation and perception are clearly separate in the agent communication space.
- Agent interaction protocols could be organised around the desired agent perception of the communication space.
- A variety of coordination principles can be embedded in the coordination media.
- State transitions can be performed in response to the occurrence of standard communication events.

Tuple centres can be programmed with reactions so as to encapsulate coordination principles directly in the coordination media. These responses are specified in terms of a reaction specification language. *TuCSoN* uses the *ReSpecT* (Reaction Specification Tuples) language [48] to specify the behaviour of the tuple centre. The behaviour of a *tuple centre* can be tailored to the application needs by defining a set of specification tuples expressed in the *ReSpecT* language. *ReSpecT* adopts a tuple language based on first order logic to defined logic tuples that are accessible via standard communication operations. Agents interact with the tuple centre using the following predicates [126], in a variant of the Linda language [74]:

- the ***out*** predicate *writes* a tuple in the tuple centre,
- the query primitives ***in, rd, inp, rdp*** provide tuple templates and expect a matching tuple back from the tuple centre, specifically:
  - ***in*** consumes the matching tuple and waits until a matching tuple is available,
  - ***inp*** consumes the matching tuple and fails if no such tuple is found,
  - ***rd*** reads and leaves the tuple in the tuple centre and waits until a matching tuple is available,
  - ***rdp*** reads and leaves the tuple in the tuple centre and fails if no such tuple is found.

Following these behaviours, a *tuple centre* can be used as a multi-agent coordination platform that provides a data-driven coordination medium through a dialogue game protocol plus full observability for agents and selective reactions over communication events.

To interact with the protocol I implemented JAVA agents that use the *TuCSoN* framework classes. The agents post tuples and the *tuple centre* reacts accordingly. So, if an invalid locution is posted, the tuple centre does not react to the locution. Known approaches to implementing dialogue protocols can be found in [53] and [10]. In [53], Doutre *et. al.* implement an Information-Seeking dialogue focused on permissions where the coordination mechanism is implemented in *TuCSoN*. In [10] Atkinson *et. al.* implement a persuasion dialogue using JAVA elements. The main difference with these approaches is that the protocol syntax and semantics in my implementation are completely embedded in the *tuple-centre*.

6.1.2 *PDGP* Syntax Specification in the Tuple Centre

The protocol syntax for the *PDGP* protocol was specified in the *tuple centre* as permanent tuples with a specific format:

$(loc(performative(parameters)))$

and the semantics rules are specified as permanent *ReSpecT* reactions. The *TuCSoN* infrastructure enables textual files for tuple centres to be created making them persistent. Each textual file contains a snapshot of the tuple centre with all the operations executed on the tuple set. This feature is used to load the protocol each time a dialogue is started. It is assumed that agents have read-only permissions for the protocol syntax tuples. Table 6.1.2 presents the syntax of the protocol in the form of *ReSpecT* tuples where the parameters used in the locutions are: (D) dialogue identifier, (Ag) agent identifier, ($Prop$) proposal identifier, (Ac) action identifier, (CqN) critical question identifier and (Ev) evidence.

Table 6.1: Some examples of *PDGP* locutions and the corresponding *ReSpecT* format. Ag_{PRO} is the proponent and Ag_{RES} is the respondent.

Locution	***ReSpecT* format**
$enter\_dialogue(Ag, D)$	$loc(enter(Ag, D))$ Agent Ag entering dialogue D.
$propose(Ag_{PRO}, D, Prop_n)$	$loc(propose(Ag, D, Prop))$ Proponent agent Ag_{PRO} proposing a plan within proposal Pr.
$accept(Ag_{RES}, D, Prop_n)$	$loc(accept(Ag, D, Prop))$ Respondent agent Ag_{RES} accepting proposal Pr
$retract(Ag_{PRO}, D, Prop_n)$	$loc(retract(Ag, D, Prop))$ Proponent agent Ag_{PRO} retracting proposal $Prop$.
$question(Ag_{RES}, D, Prop_n, CQ_n, Element)$	$loc(question(Ag, D, Prop, CqN, \_))$ Respondent agent Ag_{RES} questioning an element in $Prop$.
$assert(Ag_{PRO}, D, Prop_n, typeEvidence, \phi)$	$loc(assert(Ag, D, Prop, CqN, \_, Evidence))$ Proponent agent Ag_{PRO} asserting evidence to question CqN.

6.1.3 *PDGP* Semantics Specification in the Tuple Centre

The protocol semantics are embedded in the tuple centre as *ReSpecT* reactions. With *ReSpecT* it is possible to define rules that capture how a dialogue state changes when a locution is made. In [177], Urovi *et.al* use automated work-flows with interactions governed by a dialogue game implemented in *TuCSoN*. I use a similar approach where each reaction to a locution has rules regarding how the dialogue state changes. A reaction has the following format:

$reaction(Event, Body).$

In the "*Event*" the locution is specified and in the *Body* of the reaction the preconditions and post-conditions of the locution are implemented in the form of rules that need to be true for the reaction to be executed. So, semantics for a locution represented as a reaction are of the form:

$reaction(out(Locution1()), Preconditions, Postconditions).$

Whenever a tuple matching the template $Locution1$ is inserted in the *tuple centre*,

- a set of preconditions is checked (*e.g. Is the syntax locution valid?, Does the current dialogue stage allow the locution?*) and
- postconditions are executed (*e.g.* The dialogue history is updated, the dialogue stage changes, the agents' commitment store is updated).

The flow of a reaction for the *enter_dialogue* locution in Table 6.3 and for the *open_dialogue* is presented in Figure 6.1.

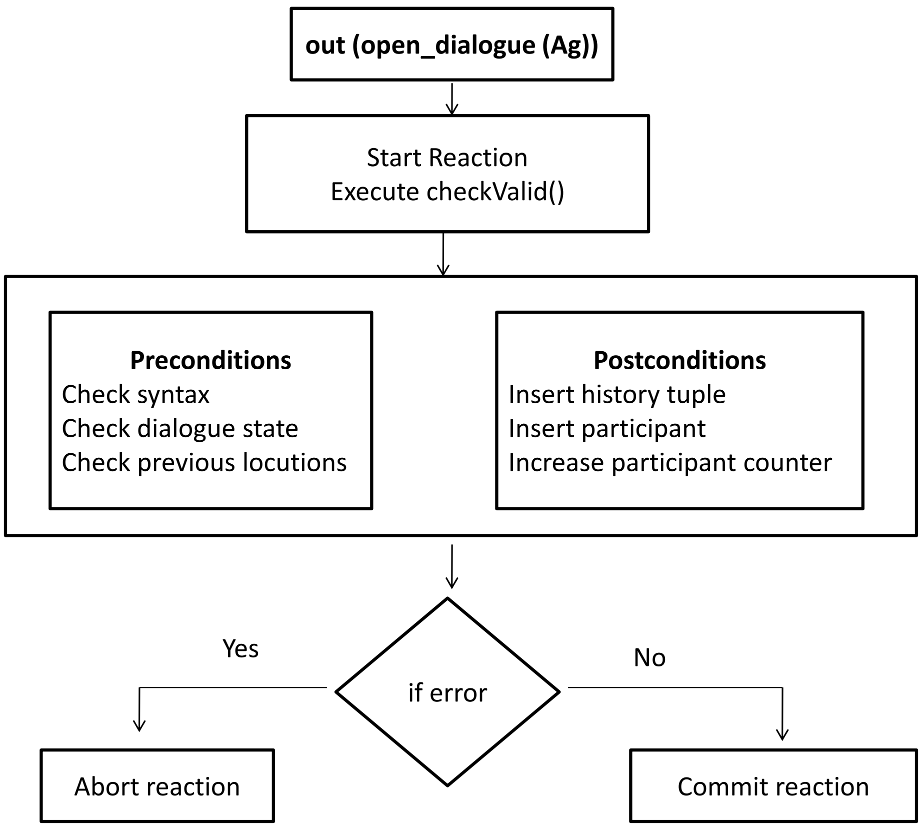


FIGURE 6.1: Diagram of the tuple centre reaction to the *open* locution

TABLE 6.2: *ReSpecT* reaction pseudo-algorithm to the *enter_dialogue()* locution.

***ReSpecT* Reaction**	**Comment**
1. out ($enter\_dialogue(Ag)$)	*Reaction specification*
2. Start reaction	
3. While no error	Start a transactional process
4. Check locution syntax	Check if there exists a tuple that matches the entry
5. Check dialogue state = idle	Validates if the dialogue state is idle
6. Insert history Tuple	Inserts a valid history tuple
7. Insert participant	Register Ag as a dialogue participant
8. *Increase participant counter*	
9. End while	
10. If Error	If there is an error in any of the reaction
11. Roll-back and Abort reaction	instructions roll-back the transaction
12. End If	
13. End Reaction	

The semantics for the $enter\_dialogue()$ and $propose()$ locutions in *ReSpecT* are given in Tables 6.3 and 6.4. The semantics of the protocol as *ReSpecT* reactions are given in Appendix B. Note that the public commitment stores are not implemented in the *ReSpecT* language.

Table 6.3: *ReSpecT* reaction for the semantics of the *enter_dialogue*() locution

***ReSpecT* Reaction**	**Comment**
$reaction(out(enter\_dialogue(Di, Ag)),$ $(out_r(checkValid(loc(enter\_dialogue(Di, Ag))))).$	*Tuple is inserted* *Check locution*
$reaction(out\_r($ $checkValid(loc(enter\_dialogue(Di, Ag)))),$	*Internal reaction to check the locution,*
Preconditions	
$rd\_r(loc(enter\_dialogue(\_, \_))),$	*If tuple is part of the syntax*
$rd\_r(dState(open)),$	*Check dialogue state*
$no\_r(dhistory(enter\_dialogue(\_, Ag))),$	*Check previous locution by same agent*
$no\_r(dhistory(open\_dialogue(\_, Ag))),$	*The dialogue should not be* *opened by the same agent*
Postconditions	
$out\_r(participant(Di, Ag)),$	*Register participant*
$in\_r(n\_participants(N)),$	*Increase the participants counter*
$N1$ is $N + 1,$	
$out\_r(n\_participants(N1)),$	
$out\_r(dhistory(enter\_dialogue(Di, Ag))),$	*Insert dialogue history tuple*
$in\_r(checkValid((\_))),$	*Clean auxiliary tuples*
$in\_r(enter\_dialogue(\_, \_))))$	

Table 6.4: *ReSpecT* reaction for the semantics of the *propose*() locution

***ReSpecT* Reaction**	**Comment**
$reaction(out(propose(Di, Ag, Pr)),$	*Propose plan tuple to react*
Preconditions	
$(rd\_r(loc(propose(\_, \_, \_))),$	*Check the locution is in the protocol syntax*
$rd\_r(dState(open)),$	*Check the dialogue state*
$rd\_r(participant(Di, Ag)),$	*Check participant Ag is in the dialogue*
Postconditions	
$out\_r(dhistory(propose(Di, Ag, Pr))),$	*Insert dialogue history tuple*
$out\_r(dState(proposing)),$	*Change dialogue state*
$in\_r(dState(open)),$	*Delete previous dialogue state*
$out\_r(role(Di, rolePr, Ag)),$	*Assign role" proponent" to the agent*
$in\_r(propose\_plan(\_, \_, \_)))).$	*Clean auxiliary tuples*

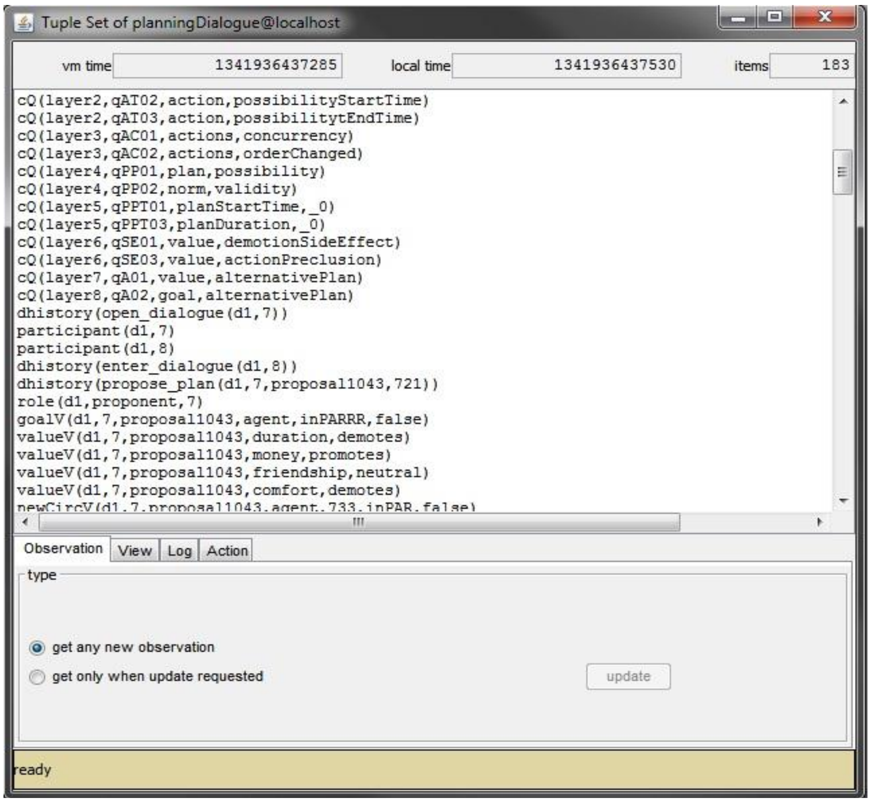
The syntax of the critical questions is the following:

$cQ(layerNumber, questionNumber, elementIdentifier, typeofAttack)$ where:

- $layerNumber$ corresponds to the critical question layer (cf. Chapter 3),
- $questionIdentifier$ refers to the critical question identifier,
- $elementIdentifier$ is the element identifier that could be the plan, action or a condition identifier,
- $typeofAttack$ refers to the type of attack.

In this way all the questions are specified in the tuple centre determining the way in which the proposal can be questioned and/or attacked. The format of the critical questions is given in Table 6.5 and Figure 6.2 presents a screen-shot of the tuple centre in the system. Figure 6.3 presents an overview of how the tuple centre is formed.

FIGURE 6.2: A snapshot of the *planningDialogue* Tuple Centre in *TuCSoN*

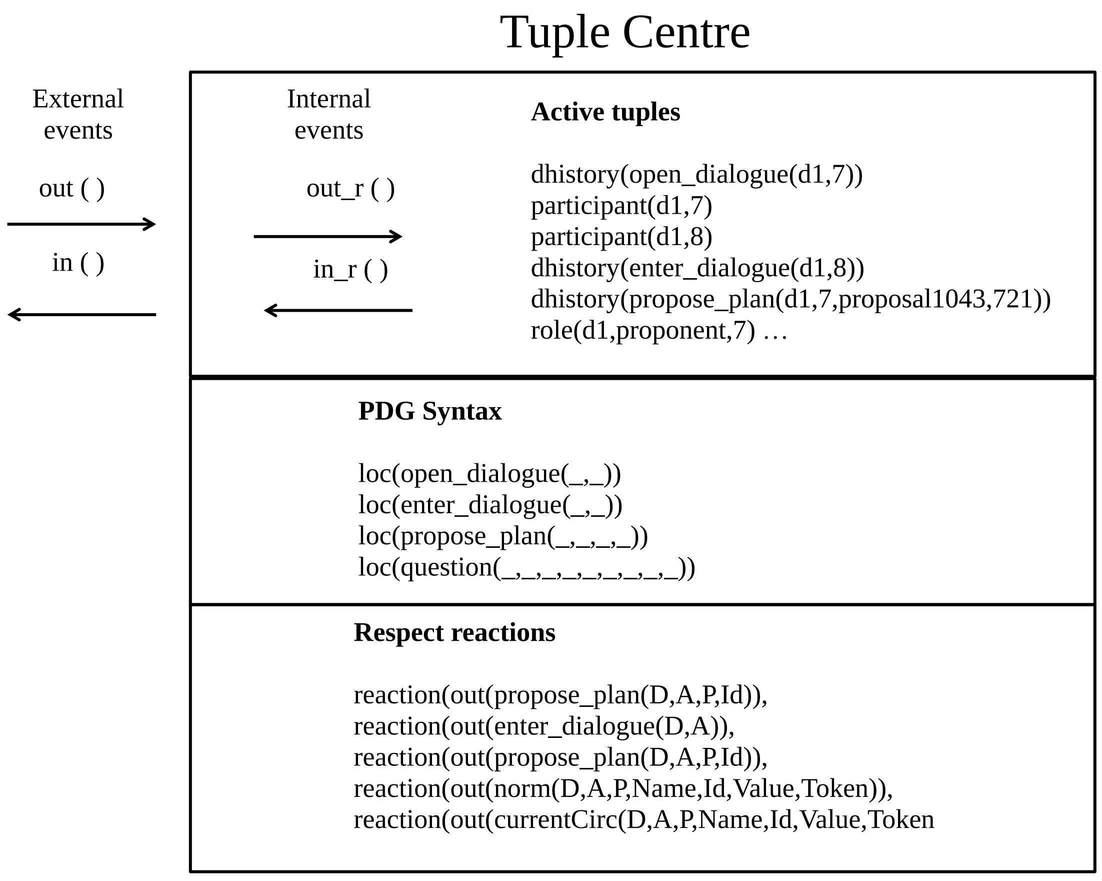

Tuple Set of planningDialogue@localhost

vm time 1341936437285 local time 1341936437530 items 183

```
cQ(layer2,qAT02,action,possibilityStartTime)
cQ(layer2,qAT03,action,possibilitytEndTime)
cQ(layer3,qAC01,actions,concurrency)
cQ(layer3,qAC02,actions,orderChanged)
cQ(layer4,qPP01,plan,possibility)
cQ(layer4,qPP02,norm,validity)
cQ(layer5,qPPT01,planStartTime,_0)
cQ(layer5,qPPT03,planDuration,_0)
cQ(layer6,qSE01,value,demotionSideEffect)
cQ(layer6,qSE03,value,actionPreclusion)
cQ(layer7,qA01,value,alternativePlan)
cQ(layer8,qA02,goal,alternativePlan)
dhistory(open_dialogue(d1,7))
participant(d1,7)
participant(d1,8)
dhistory(enter_dialogue(d1,8))
dhistory(propose_plan(d1,7,proposal1043,721))
role(d1,proponent,7)
goalV(d1,7,proposal1043,agent,inPARRR,false)
valueV(d1,7,proposal1043,duration,demotes)
valueV(d1,7,proposal1043,money,promotes)
valueV(d1,7,proposal1043,friendship,neutral)
valueV(d1,7,proposal1043,comfort,demotes)
newCircV(d1,7,proposal1043,agent,733,inPAR,false)
```

Table 6.5: Critical Questions *ReSpecT* format

Critical Questions Examples	**Description**
cQ(layer1,qA06,startEffects,possibility)	*Layer 1, question CQA-06, questions the possibility of a start effect of an action*
cQ(layer2,qAT02,startTime,possibility)	*Layer 2, question CQAT-02, questions the possibility of the start time of an action*
cQ(layer3,qAC01,Action1,Action2,concurrency)	*Layer 3, question CQAC-01, questions concurrency of two actions*
cQ(layer4,qPP02,norm,validity)	*Layer 4, question CQPP-02, questions the validity of a norm*
cQ(layer5,qPPT01,startTime,possibility)	*Layer 5, question CQPT-01, questions the possibility of the start time of the plan*
cQ(layer6,qSE01,value,demotionSideEffect)	*Layer 6, question CQSE-01, questions demotion of a value*
cQ(layer7,qAO01,value,alternativePlan)	*Layer 7, question CQAO-01, inquires for an alternative plan to promote the same value*



Tuple Centre
External events
out ( )
in ( )
Internal events
out_r ( )
in_r ( )
Active tuples
dhistory(open_dialogue(d1,7))
participant(d1,7)
participant(d1,8)
dhistory(enter_dialogue(d1,8))
dhistory(propose_plan(d1,7,proposal1043,721))
role(d1,proponent,7) …
PDG Syntax
loc(open_dialogue(_,_))
loc(enter_dialogue(_,_))
loc(propose_plan(_,_,_,_))
loc(question(_,_,_,_,_,_,_,_,_))
Respect reactions
reaction(out(propose_plan(D,A,P,Id)),
reaction(out(enter_dialogue(D,A)),
reaction(out(propose_plan(D,A,P,Id)),
reaction(out(norm(D,A,P,Name,Id,Value,Token)),
reaction(out(currentCirc(D,A,P,Name,Id,Value,Token

Figure 6.3: Protocol elements in the *planningDialogue* Tuple Centre

6.2 Agent and System Architecture

The high level architecture description is as follows: agents have two main modules, a *dialogue manager* and a *planning engine.* I used the *dialogue manager* and *planning engine* concepts from the TRAINS implementation presented in [2] where a conversational planning agent engages in a dialogue to create a plan, using feedback received from the interaction.

The dialogue manager is in charge of creating the planning arguments and deals with the plan proposals through the protocol embedded in the Tuple centre. The dialogue manager is embedded in the agent architecture and mainly handles plan proposals and arguments generated from the dialogue. Through the dialogue manager agents can:

- Create plan proposals.
- Create valid tuples to engage in a dialogue.
- Apply the critical question selection strategy.
- Host the interface to communicate with the tuple centre to:
    - Post agents' locutions in the form of valid tuples.
    - Retrieve the protocol syntax.
    - Retrieve critical questions.
    - Retrieve dialogue participants.
    - Retrieve the dialogue history.
    - Retrieve the proposal details.
    - Retrieve the commitments.

The *planning engine* has the following capabilities:

- Select the preferred plan from the plan library based on the agent's current preference.
- Hold the interface with the dialogue manager

Figure 6.4 gives an overview of the agent components and the system architecture. The strategy to select questions in the dialogue (as presented in Chapter 5) is separated from the dialogue manager to give the agents flexibility to change the strategy according to different scenarios without building the strategy into the dialogue manager itself. Figure 6.5 presents an overview of how the strategy works within the system and agents' architecture.

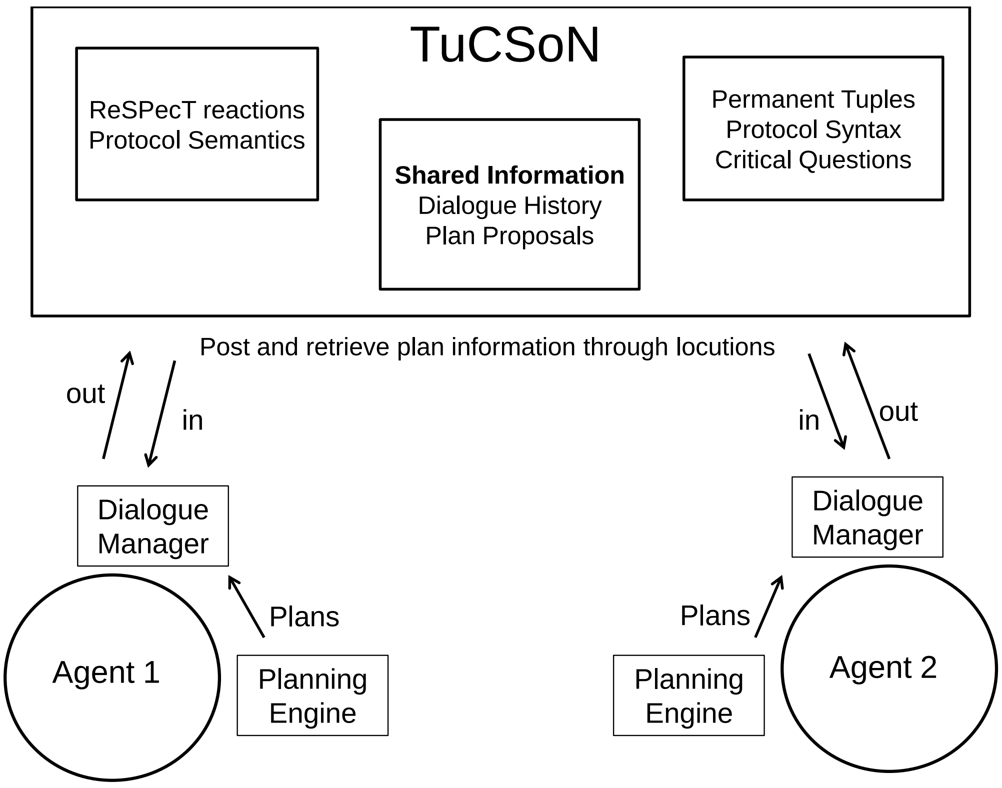


Figure 6.4: Agent and system architecture.

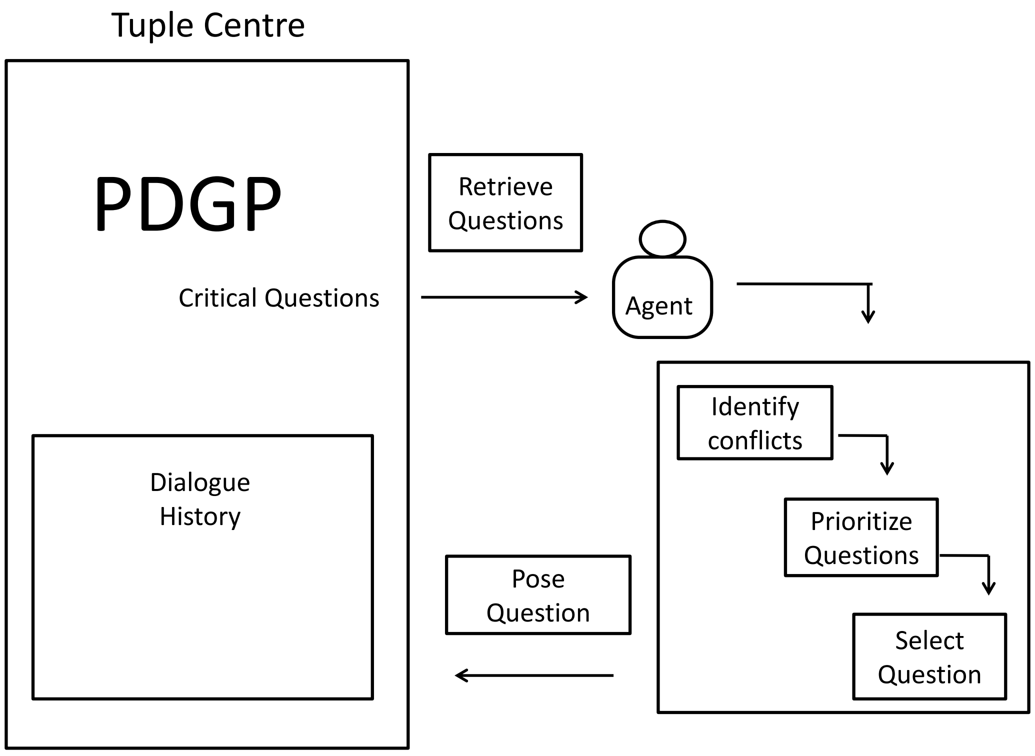


Figure 6.5: Strategy within the agent and system architecture.

Typically, a multi-agent planning task requires a description of the initial state, a set of action capabilities and a set of private goals. The context representation is based in the *PDDL* specification, the Planning Domain Definition Language [76]. A *PDDL* specification consists of a set of predicates, a set of actions with parameters and a set of preconditions and effects for each action. *PDDL 2.1* [71] introduced the concept of durative actions (as presented in Chapter 3) allowing the action duration to be included together with more conditions and effects.

In *PDDL 2.1* a condition can be a precondition (*at_start*), an invariant condition (*over_all*) or a termination condition (*at_end*) and action effects can be start effects (*at_start*) or end effects (*at_end*). These action elements are used to represent the state of the world and the actions in Java classes.

6.3 Design of the Test Cases

6.3.1 Journey Example

The implementation is based scenario where agents need to agree on a plan to attend a conference in Paris travelling from Inverness. This example treats multi-agent plans in a different way as the way they were printed in Chapter 3. Agents execute the same plan but still they may have different beliefs about its specification. The situation is the following: two agents (named John and Paul) both have different plans to reach Paris and different beliefs about the current circumstances.

The actions to travel are: travel by train *takeTrain*(), take a flight *takeFlight*() and take a coach *takeCoach*() through four cities Inverness, Manchester, London and Paris. I assume that both agents execute the same action i.e. John takes a train and Paul takes a train. The actions are the following:

- $takeCoach(Inverness, Manchester)$
- $takeCoach(Manchester, London)$
- $takeCoach(London, Paris)$
- $takeTrain(Inverness, Manchester)$
- $takeTrain(Manchester, Paris)$,
- $takeFlight(Inverness, Paris)$
- $takeFlight(London, Paris)$

Each city has restrictions on the availability of the train station, airport and coach station.The values that can be associated with the plans are:

- Value $v_1 = money$, the cheapest option.
- Value $v_2 = time$, the fastest option.
- Value $v_3 = friendship$, agents travelling together.
- Value $v_4 = comfort$, the most comfortable way to travel.

Figure 6.6 presents the available actions that agents can choose with the restrictions on each city. Table 6.6 presents some the possible plans agents can propose. The following sections focus on the problem of creating a proposal taking into account all the details that the problem poses and how these proposals could be challenged.

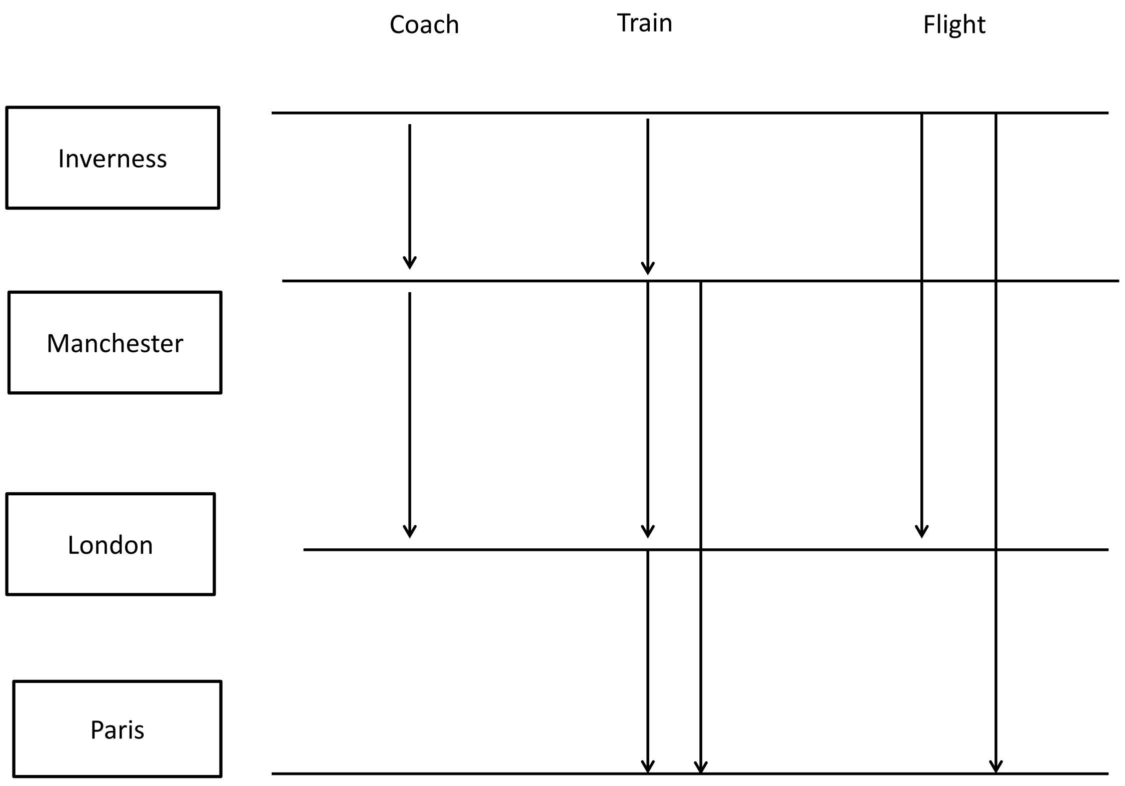


Figure 6.6: Journey Example. Possible connections between cities.

Table 6.6: Agents' plans for the Journey Example

Plan	**Actions**	**Values**	
PL_1 - Coach	$takeCoach(Inverness, Manchester)$ $takeCoach(Manchester, London)$ $takeCoach(London, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$+$ $-$ $=$ $-$
PL_2- Trains	$takeTrain(Inverness, Manchester)$ $takeTrain(Manchester, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$=$ $=$ $+$ $+$
PL_3 - Flight	$takeFlight(Inverness, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$-$ $+$ $-$ $=$
PL_4 - Coach-Train	$takeCoach(Inverness, Manchester)$ $takeTrain(Manchester, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$=$ $-$ $+$ $=$
PL_5 - Train-Flight	$takeTrain(Inverness, London)$ $takeFlight(London, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$-$ $=$ $=$ $=$
PL_6 - Coach-Train-Flight	$takeCoach(Inverness, Manchester)$ $takeTrain(Manchester, London)$ $takeTrain(London, Paris)$	$v_1 = money$ $v_2 = duration$ $v_3 = friendship$ $v_4 = comfort$	$-$ $=$ $=$ $-$

The test cases are formed by providing the agents with:

1. A set of propositions that represent constraints of the "social context".
2. An initial view of the world.
3. A set of action specifications.
4. A set of plans.
5. A set of values.
6. A preference order over values.

Agents' plans are presented in Table 6.6 together with the status of the values (promoted (+), demoted (−) or neutral (=)). Actions are meant to be executed simultaneously by both agents. Although each individual action could be associated with a value, for the sake of simplicity here I will only consider values related to the plan as a whole.

The validity of some elements in the plans is changed for the different test cases and/or world representation for each agent to create different scenarios. The validity of elements (actions conditions, action effects, norms) is represented using a "token attribute" associated with each element. That the "token" is $false$ represents that the element validity against the context has expired. The evaluation of arguments that attack the proposal is done automatically through the token functionality.

Agents are pre-loaded four different sets of information about the world and plans (presented in Table 6.7). These sets are then combine them with five different preference orders for the agents to generate twenty test cases. In Table 6.7, a check mark (✓) indicates the validity of the element and a cross (×) indicates that an invalid element exists in the specification. The twenty test cases constructed with information in Tables 6.7 and 6.8 to generate dialogue runs. The test cases are described below:

- Test case A: All the plans and information about the world is valid for both agents. Only questions suggesting alternative plans are generated for this test case. Agents focus on selecting the best option on each run.
- Test case B: John's plans PL_1, PL_2, PL_3 are invalid inducing questions about the validity and possibility of the action elements. Any preference over these plans has to be re-evaluated after the dialogue. The social constraints and initial state of Paul are not valid and generate possibility questions for norms and suitability questions regarding the initial state.
- Test case C: John's plans PL_2, PL_3 are invalid. Again any preference over these plans has to be re-evaluated after the dialogue. Paul's plan PL_4 is also invalid.

TABLE 6.7: Test case specifications

Test case	**John's Plans**		**John's Beliefs**		**Paul's Plans**		**Paul's Beliefs**	
A	PL_1 PL_2 PL_3 PL_6	✓ ✓ ✓ ✓	Social constraints Initial state Action specification	✓ ✓ ✓	PL_4 PL_5	✓ ✓	Social constraints Initial state Action specification	✓ ✓ ✓
B	PL_1 PL_2 PL_3 PL_6	× × × ✓	Social constraints Initial state Action specification	✓ ✓ ✓	PL_4 PL_5	✓ ✓	Social constraints Initial state Action specification	× × ✓
C	PL_1 PL_2 PL_3 PL_6	✓ × × ✓	Social constraints Initial state Action specification	✓ ✓ ✓	PL_5 PL_4	✓ ×	Social constraints Initial state Action specification	✓ ✓ ✓
D	PL_1 PL_2 PL_3 PL_6	✓ × ✓ ✓	Social constraints Initial state Action specification	✓ × ✓	PL_4 PL_5	✓ ✓	Social constraints Initial state Action specification	✓ ✓ ✓

- Test case D: John's initial state is not valid nor is his plan PL_2. This case aims to question the proposals mainly at the plan suitability level.

Table 6.8 presents the five sets of preference orders used for both agents. Agents put forward first their preferred plans and proceed in descending order of priority. Agents may change their preferred plan during the dialogue depending on the outcome of the questioning process. More details on how the agents' preferred plan changes after the dialogue are given in the results presented in the next chapter. Each test case is executed through a dialogue simulation six times to test the different strategies used (the two strategies from Chapter 5 and a random approach) and the agent that starts the dialogue (two agents), since which agent goes first might have a significant influence on the dialogue. Table 5.2 in the previous chapter presents the order in which questions are considered for both strategies.

Table 6.8: Test cases preferences and the preferred plan for both agents.

	John preference		**Paul preference**	
Test case	**Value preference order**	**Preferred value in plan**	**Value preference order**	**Preferred value in plan**
1	$v_1 > v_4 > v_3 > v_2$	v_1 in PL_1	$v_3 > v_4 > v_1 > v_2$	v_3 in PL_4
2	$v_3 > v_2 > v_1 > v_4$	v_3 in PL_2	$v_3 > v_4 > v_1 > v_2$	v_3 in PL_4
3	$v_2 > v_1 > v_3 > v_4$	v_2 in PL_1	$v_2 > v_4 > v_1 > v_3$	v_2 in PL_5
4	$v_3 > v_1 > v_4 > v_2$	v_3 in PL_2	$v_1 > v_2 > v_4 > v_3$	v_1 in PL_4
5	$v_1 > v_4 > v_3 > v_2$	v_1 in PL_1	$v_2 > v_4 > v_3 > v_1$	v_3 in PL_4

From the list critical questions in Section 3.5, twenty eight questions were implemented for these experiments. I implemented critical questions related to validity, possibility and suitability for the action and plan specifications, alternative plans and side effect questions. Critical questions regarding action-combination and time-related questions to avoid the complexity of implementing those characteristics for the example.

6.4 Dialogue Runs

A dialogue-run consists of the following steps:

1. A proponent agent initiates the dialogue.
2. The agent selects the most preferred plan according to its value preference order.
3. The dialogue manager transforms the plan into a proposal object.
4. The dialogue manager creates a valid tuple using the proposal object.
5. The protocol in the tuple centre validates the locution.
6. The respondent agent acknowledges the proposal and starts the questioning process, applying the strategy configured to the agent.

7. The questioning process involves the respondent agent questioning the proponent over the elements in the proposal until acceptance, retraction or rejection of the proposal.

8. When the respondent agent accepts a proposal the questioning process stops and an alternative option is evaluated through question *CQAO-03: Is there an alternative plan to realise the same goal?*, the agents change roles.

   The respondent agent then gets a turn to propose and evaluate its preferred plan (note that the algorithm just give the respondent agent one chance to put forward a preferred plan).

9. Once the questioning process finishes, if the preferred plan for both is the same the dialogue finishes. After the agents have accepted on whether a plan is in principle unquestionable, they have to agree on one, based on the preferences of the agents. Note that in the case of two possible plans at the end of the dialogue, the proponent selects the plan which promotes more values, or when these are equal, the plan that demotes fewer values. Further details on how the dialogue simulation works are given in the next chapter when the results are analysed.

In the dialogue, critical questions are used to attack an element of the argumentation scheme through the challenge locution. For the purpose of the implementation it is sufficient to formulate the question as an attack.

Following the definition of agreeable and disagreeable agents used in [192] an agreeable agent accepts arguments proposed by other participant agents and only promotes its own argument when an agreement is not possible. A disagreeable agent only accepts an argument when it is forced to do so, because it has no possible counterargument. Thus an agreeable agent gives the “benefit of the doubt” when it is unsure of a piece of information.

The agents implemented are disagreeable in the sense that they only accept arguments from an opponent when there is no reason to reject them and the respondent agent always get the chance to promote its own argument. If there is no agreement on a plan when all the questions have been posed the proponent agent selects the best option. Agents do not necessarily agree on a plan they find acceptable since there could be side effects and alternative options that need to be evaluated before agreeing on a plan.

Since I use a Java implementation, the communication between modules is done through objects. The pseudo-code in Figure 6.7 presents how the agents create and interchange objects in a dialogue run from the point when a proposal needs to be presented. Figure 6.7 shows the pseudo-code that generates the dialogue simulations. Examples of the result of dialogue runs are given in Appendices C and D.

Note that if the current state, and the initial proposal for a plan are given the outcome of the dialogue is not determined and will depends on the beliefs and preferences of the participants.

6.5 Chapter Summary

This chapter presents the computational implementation of simulated dialogue runs using the concepts presented in Chapters 3, 4 and 5.

The implementation covers the *PDGP* dialogue protocol, the agent and system architecture, the design of the test cases to create the dialogue simulations and the dialogue simulations themselves. The implementation of the *PDGP* protocol is done using *TuCSoN*, a blackboard software architecture and the protocol semantics are implemented in the *ReSpecT* language. The agent and system are implemented in Java and use *TuCSoN* as the medium to communicate.

The test cases conceptualize agents with different information about the world, different plans and different preferences. Agents use the argumentation scheme for plan proposals *ASP* of Chapter 3 to present their plans and the related critical questions are part of the protocol in the tuple centre. The dialogue simulations are based on twenty test cases where two agents critique each other's plans.

The dialogue begins with a plan proposal and questions are put forth over supposed conflicts over elements in the proposal. The strategies presented in Chapter 5 were also implemented as configurable specifications that agents can use when questioning a plan. In the next chapter the results of these experiments are presented and analysed

```
Proponent Agent

01.  ProposePlan()
          \\Agent selects the preferred plan
02.      PE.SelectPreferredPlan()
          \\Create proposal from object plan
03.      DM.CreateProposal()
          \\Pose tuple proposal to the tuple centre
04.      DM.PoseProposal()
05.      TucsonContext.PostTuple()
          \\Proponent waits for questions
06.      DialogueThread.WaitforQuestions()
07.      If PlanQuestioned()
08.         For each question
                \\Search for evidence related to question
09.             DM.SearchEvidence()
10.           If EvidenceFound()
                    \\Provides evidence related to the question
11.                 DM.ProvideEvidence()
12.           else
13.                 DM.RetractProposal()
14.           End if
15.       Next
16.    End If
17. End

Respondent Agent

01. DialogueThread.QuestionProposal()
        \\Obtain configured strategy
02.    DM.ApplyStrategy()
           \\Identify valid questions
03.       DM.IdentifyConflicts()
           \\Prioritize questions based on configured strategy
04.       DM.Orderquestions(strategy)
05.       While QuestionList.haselements()
              \\Pose question to the tuple centre
06.           DM.PoseNextQuestion()
                \\Wait from proponent answer
07.            DialogueThread.WaitforAnswer()
08.          If EvidencetProvided()
09.                 DM.DismissQuestion
10.             else
                     \\Question defeats proposal and break the loop
11.                  break()
12.            End If
13.       End While
14.   End
```

Figure 6.7: Pseudo-code for the proponent and respondent agents' functions in a dialogue simulation.

```
1. For each plan in the proponent plan repository
2. Identify the preferred plan
3.      Add to list of plans to pose
4. End For
5.
6. While list of plans not empty
7.     Propose plan
8.     Evaluate with Critical Questions
9.     If Plan accepted
10.          Exit Loop
11.    else \\ Plan rejected
12.     Remove plan from list
13.     Continue Evaluation with the next plan
14.    End if
15. End While
16.
17. For all respondent options found
18.     Propose the preferred plan
19.   Evaluate proposal
20. End For
```

Figure 6.8: Dialogue simulation pseudo-code.

Chapter 7

Evaluation and Analysis of Results

This chapter presents an analysis of the simulated dialogue runs described in the previous chapter. The results indicate a largely positive assessment of the approach and provide several entry points for discussion and future work. Strategies are evaluated on the number of questions the dialogue needed to end the dialogue. The outcome of the dialogues (i.e. the plan agreed upon) does not play a role in measuring the efficiency of the strategies but it does reflect the selection of an possible plan for both agents based on their differences in beliefs and preferences.

This chapter is structured as follows: the results of the dialogue simulations are presented in Section 7.1 with four subsections (7.1.1 - 7.1.2) where the output of the dialogue simulations for each test case is explained and discussed. Section 7.2 presents a summary of the results with an emphasis on the positive and negative aspects of the implementation.

7.1 Dialogue Simulation Results

The results of the dialogue simulations are presented in Tables 7.1 - 7.8. Some examples of the actual simulation outputs are given in Appendices C and D. The results of each test case are presented in separate tables where each table contains five sets of results representing a different agent's preference. A result-table is composed of the results of the simulation of a test case (A,B,C,D) (see Table 6.7) together with a set of agent's preference (see Table 6.8) that forms the test case combination represented by: the test case letter and the preference number (e.g. A1, A2, etc.).

Each combination was executed four times, where agents use strategy $s1$, strategy $s2$, (see Table 5.2), strategy $s3$ which takes a random question selection and a scenario where there is no strategy (NoSt) that is, no question identification and no question prioritisation.

These dialogue simulations are then repeated changing the agent that starts the dialogue, to obtain forty (i.e. 1 test case $\times$ 4 strategies $\times$ 5 agent preferences $\times$ 2 agent that starts) results per test case presented in two tables. To analyse the results, the tables present:

1. The number of proposals evaluated (each proposal contains one plan).
2. The plans put forward by the *proponent* in the order they were evaluated. The proposals are presented in the form:

   [*Number of proposals – Proponent agent: Plan evaluated (outcome),.., Plan evaluated (outcome)*]
3. The outcome of the individual plan evaluation. This outcome is presented as, either a check mark (✓) for a proposal accepted by the *respondent* or a cross ($\times$) for a rejected proposal. A rejected proposal indicates that the *proponent* was not able to defend successfully the argument for the plan that is, it was defeated with the use of a critical question. In this case, questions are presented as attacks and therefore defeat the arguments for plan proposals but other evaluation semantics could be used (cf. to Section 6.3 for a detailed account on how questions defeat proposals.).
4. The overall number of questions (No.Q.) issued by both participants.
5. The plan agreed upon.

The value preference of the agents before and after the dialogue is also presented. In the following subsections I present the results and comment on the outcome of each test case.

7.1.1 Results from Dialogue Simulations of Test Case A

In test case A neither plans or agents' beliefs have inconsistencies. The results are presented in Tables 7.1 and 7.2. The results from test cases A1 and A5 from Table 7.1 are presented with a thorough explanation that explains each element of the table. For the rest of the test cases I will provide general comments on the results.

Explanation of test case A1 (Table 7.1):

- John starts the dialogue with a preference for plan PL_1. Plan PL_1 promotes the value $v_1 = money$ (cf. Table 6.6) and preference 1 (cf. Table 6.8) gives priority to v_1.
- Paul prefers plan PL_4 because it promotes value $v_3 = friendship$ (cf. Table 6.6) and even though in preference 1 (cf. Table 6.8) Paul gives priority to v_4, there is no plan in its repertoire that promotes v_4 so he gives priority to the next value, v_3 in this case.

- In the first run, Paul uses strategy $s1$ to pose questions and the simulation evaluates two proposals. John proposes first plan PL_1 which is accepted (✓) by Paul.
- The proponent role of John finishes and Paul, through question *CQAO-05: Is there an alternative better plan to reach the goal?*, gets a turn to propose and evaluate plan PL_4 which is also accepted. The proposing dialogue stage thus ends with two possible plans. John finally chooses plan PL_4 because it demotes fewer values overall (cf. step 9 in Section 6.4).
- The dialogue finishes then with one question posed and plan PL_4 selected.
- For run 2, the ordering of the critical questions changes since strategy $s2$ is used. In run 2 plans are evaluated in the same order as in run 1. The strategy used to select critical questions should not affect the outcome of the dialogue, but just influence the length of the dialogue.
- Run 3 also follows the same order and evaluation of proposals as runs 1 and 2 but since no strategy is used Paul poses all the questions, because the random approach has no process to identify relevant questions. The plan selected is the same but the number of questions issued between participants is 119. The number of questions depends largely on the number of conditions and effects that the proposal presents.

Explanation of test case A5 (Table 7.1):

- John starts the dialogue having preference for plan PL_1. Plan PL_1 promotes the value $v_1 = money$ (cf. Table 6.6) and John's preference 5 (cf. Table 6.8) gives priority to this value. Paul prefers plan PL_5 even though it does not promote v_2 (his preferred value) because PL_4 demotes v_2.
- In the first proposal, run 1, Paul uses strategy $s1$ to pose questions. The simulation evaluates three proposals. John proposes first plan PL_1 which is rejected by Paul because of its side effects contained in question *CQSE-01: Does the plan PL_1 have a side effect which demotes the value v_n?* Plan PL_1 demotes value $v_2 = duration$ which is Paul's highest ranked value. The simulation evaluates three questions:
  1. Paul asks: *CQSE-01: Does the plan PL_1 have a side effect which demotes the value v_n?* The proposal for PL_1 is rejected.
  2. Paul asks: *CQAO-03: Is there an alternative better plan to reach the goal?*
  3. John asks: *CQSE-01: Does the plan PL_5 have a side effect which demotes the value v_n?* The proposal for PL_5 is rejected.
- The second proposal (plan PL_2) is accepted by Paul since no possible attacks on John's proposal are identified.

- The third proposal (plan PL_5) made by Paul is rejected because of the same side effect question (*CQSE-01*). Plan PL_5 does not promote any value but demotes $v_1 = \mathit{money}$, John's highest ranked value in preference 5 (cf. Table 6.8).

I present now some observations on the results of test case A:

- I use this test case as a validation of the first step in the strategy related to the belief alignment of agents. Given that the agents' representation of the world is the same, agents do not identify question related to the validity and possibility of the elements presented and therefore, no questions are added to the list.
- Neither plans nor agents' beliefs have inconsistencies so all the strategies pose just one critical question: *CQAO-05. Is there an alternative better plan to reach the goal?* See Section 6.4 to check details on how the dialogue runs are implemented.
- For this test case where no inconsistencies were found there is no difference in the results when the agent that starts the dialogue is changed.
- In all the runs there are two possible plans at the end of the evaluation. The proponent agent choose the plan which promotes more values, or when these are equal, the plan that demotes fewer values. In run A1, after plans PL_1 and PL_4 are evaluated, plan PL_4 demotes fewer values than plan PL_1 and so PL_4 is chosen as the agreed plan.
- The outcome of the dialogue and the order of the proposals does not change when the strategy is changed, but the number of questions increases considerably when using the random approach.

TABLE 7.1: Test case A when John starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
A1	John: $PL_1 \rightarrow PL_1$	$s1$	2 - J: PL_1(✓) P: PL_4(✓)	1	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	2 - J: PL_1(✓) P: PL_4(✓)	1	PL_4
		$s3$	2 - J: PL_1(✓) P: PL_4(✓)	1	PL_4
		NoSt	2 - J: PL_1(✓) P: PL_4(✓)	119	PL_4
A2	John: $PL_2 \rightarrow PL_2$	$s1$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
	Paul: $PL_4 \rightarrow PL_4$	$s2$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
		$s3$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
		NoSt	2 - J: PL_1(✓) P: PL_4(✓)	169	PL_2
A3	John: $PL_3 \rightarrow PL_3$	$s1$	2 - J: PL_3(✓) P: PL_5(✓)	1	PL_3
	Paul: $PL_5 \rightarrow PL_5$	$s2$	2 - J: PL_3(✓) P: PL_5(✓)	1	PL_3
		$s3$	2 - J: PL_3(✓) P: PL_5(✓)	1	PL_3
		NoSt	2 - J: PL_3(✓) P: PL_5(✓)	91	PL_3
A4	John: $PL_2 \rightarrow PL_2$	$s1$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
	Paul: $PL_4 \rightarrow PL_4$	$s2$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
		$s3$	2 - J: PL_2(✓) P: PL_4(✓)	1	PL_2
		NoSt	2 - John: PL_2(✓) P: PL_4(✓)	169	PL_2
A5	John: $PL_1 \rightarrow PL_2$	$s1$	3 - J: PL_1(×),PL_2(✓) P: PL_5(×)	3	PL_2
	Paul: $PL_5 \rightarrow -$	$s2$	3 - J: PL_1(×),PL_2(✓) P: PL_5(×)	3	PL_2
		$s3$	3 - J: PL_1(×),PL_2(✓) P: PL_5(×)	3	PL_2
		NoSt	3 - J: PL_1(×),PL_2(✓) P: PL_5(×)	148	PL_2

Table 7.2: Test case A when Paul starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
A1	John $PL_1 \rightarrow PL_1$	$s1$	2 - P: PL_4(✓). J: PL_1(✓)	1	PL_4
	Paul $PL_4 \rightarrow PL_4$	$s2$	2 - P: PL_4(✓). J: PL_1(✓)	1	PL_4
		$s3$	2 - P: PL_4(✓). J: PL_1(✓)	1	PL_4
		NoSt	P: 2 - PL_4(✓). J: PL_1(✓)	119	PL_4
A2	John $PL_2 \rightarrow PL_2$	$s1$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
	Paul $PL_4 \rightarrow PL_4$	$s2$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
		$s3$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
		NoSt	2 - P: PL_4(✓). J: PL_2(✓)	169	PL_2
A3	John $PL_3 \rightarrow PL_3$	$s1$	2 - P: PL_5(✓). J: PL_3(✓)	1	PL_3
	Paul $PL_5 \rightarrow PL_5$	$s2$	2 - P: PL_5(✓). J: PL_3(✓)	1	PL_3
		$s3$	2 - P: PL_5(✓). J: PL_3(✓)	1	PL_3
		NoSt	2 - P: PL_5(✓). J: PL_3(✓)	91	PL_3
A4	John $PL_2 \rightarrow PL_2$	$s1$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
	Paul $PL_4 \rightarrow PL_4$	$s2$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
		$s3$	2 - P: PL_4(✓). J: PL_2(✓)	1	PL_2
		NoSt	2 - P: PL_4(✓). J: PL_2(✓)	169	PL_2
A5	John $PL_1 \rightarrow -$	$s1$	3 - P: $PL_5(\times)$,PL_4(✓). J: $PL_1(\times)$	3	PL_4
	Paul $PL_5 \rightarrow PL_4$	$s2$	3 - P: $PL_5(\times)$,PL_4(✓). J: $PL_1(\times)$	3	PL_4
		$s3$	3 - P: $PL_5(\times)$,PL_4(✓). J: $PL_1(\times)$	3	PL_4
		NoSt	3 - P: $PL_5(\times)$,PL_4(✓). J: $PL_1(\times)$	148	PL_4

7.1.2 Observations from Dialogue Simulations of Test Case B

I present now observations on the results of test case B (Tables 7.3 and 7.4).

-

- When John starts the dialogue proposal plan PL_6 is always selected since it is the only valid plan.

- In run B1 (Table 7.3), agents evaluate five proposals. Even though plans PL_4 and PL_5 are valid plans, the norms and initial state are not valid and that is why the proposals are rejected.

- In run 2, agents evaluate three proposals. First, plan PL_2 proposal is rejected by Paul, the next value preferred for John is v_2 so John picks PL_6 which is neutral to v_2 (plan PL_1 is also neutral to $v_2 = duration$). Finally ,Paul proposes plan PL_4 through the alternative question option; the proposal is rejected and the proposal process stops.

- Strategy $s2$ performs better (number of questions exchanged) in the first four orderings (B1-B4) because most of the inconsistencies were induced in the possibility for action conditions in the plans' representations which strategy $s2$ considers first. As discussed previously, strategy $s1$ puts forward suitability questions first, followed by validity and possibility questions.

- When Paul starts the dialogue (Table 7.4) there is no outcome since none of the plans presented is valid. Although plan PL_4 and plan PL_5 are valid plans the norms and initial state of Paul are not valid and this is the reason why the proposals get rejected. Agent John does not get the chance to present plan PL_6 because the plan is not preferred in any run. The dialogue implementation is not fair in that sense. Ideally, there should be no difference as to who starts the dialogue. The simulation gives the "respondent agent" just one opportunity to pose its best-preferred plan, and if that plan turns out to be not valid there is no second chance. The proponent agent has the opportunity to pose all its plans until one is accepted.

- In all the runs for this test case agents do not have a preference after the dialogue since their initial preferences were discarded and the plan selected does not promote their first option. Nevertheless, the best-preferred plan is selected (in this test case is this not evident since only one plan is valid in the final evaluation).

  Regarding the random ordering $s3$ show that when using either strategy s1 or s2 the result in terms on number of questions exchanged. In all the cases strategies s3 and s2 perform better than the random ordering in most of the cases.

Table 7.3: Test case B when John starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
B1	John: $PL_1 \rightarrow -$	$s1$	5- J: $PL_1, PL_2, PL_3(\times)$, $PL_6(\checkmark)$ P: $PL_4(\times)$	23	PL_6
	Paul: $PL_4 \rightarrow -$	$s2$	5- J: $PL_1, PL_2, PL_3(\times)$,$PL_6(\checkmark)$ P: $PL_4(\times)$	12	PL_6
		$s3$	5- J: $PL_1, PL_2, PL_3(\times)$,$PL_6(\checkmark)$ P: $PL_4(\times)$	21	PL_6
		NoSt	5- J: $PL_1, PL_2, PL_3(\times)$, $PL_6(\checkmark)$ P: $PL_4(\times)$	184	PL_6
B2	John: $PL_2 \rightarrow -$	$s1$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	15	PL_6
	Paul: $PL_4 \rightarrow -$	$s2$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	9	PL_6
		$s3$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	14	PL_6
		NoSt	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	150	PL_6
B3	John: $PL_3 \rightarrow -$	$s1$	3- J: $PL_3(\times), PL_6(\checkmark), P : PL_5(\times)$	13	PL_6
	Paul: $PL_5 \rightarrow -$	$s2$	3- J: $PL_3(\times), PL_6(\checkmark), P : PL_5(\times)$	8	PL_6
		$s3$	3- J: $PL_3(\times), PL_6(\checkmark), P : PL_5(\times)$	12	PL_6
		NoSt	3- J: $PL_3(\times), PL_6(\checkmark), P : PL_5(\times)$	168	PL_6
B4	John: $PL_2 \rightarrow -$	$s1$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	15	PL_6
	Paul: $PL_4 \rightarrow -$	$s2$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	9	PL_6
		$s3$	3- J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	20	PL_6
		NoSt	3 - J: $PL_2(\times), PL_6(\checkmark), P : PL_4(\times)$	150	PL_6
B5	John: $PL_1 \rightarrow -$	$s1$	5- J: $PL_1, PL_2, PL_3(\times), PL_6(\checkmark)$ P: $PL_5(\times)$	15	PL_6
	Paul: $PL_5 \rightarrow -$	$s2$	5- J: $PL_1, PL_2, PL_3(\times), PL_6(\checkmark)$ P: $PL_5(\times)$	23	PL_6
		$s3$	5- J: $PL_1, PL_2, PL_3(\times), PL_6(\checkmark)$ P: $PL_5(\times)$	21	PL_6
		NoSt	5 - J: $PL_1, PL_2, PL_3(\times), PL_6(\checkmark)$ P: $PL_5(\times)$	139	PL_6

Table 7.4: Test-case B when Paul starts the dialogue

Test case	**Preferred plan before $\rightarrow$ after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
B1	John: $PL_1 \rightarrow -$	$s1$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_1(\times)$	10	-
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times)$, $PL_5(\times)$, $J : PL_2(\times)$	4	-
		$s3$	3 - P: $PL_4(\times)$, $PL_5(\times)$, $J : PL_2(\times)$	10	
		NoSt	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	157	-
B2	John: $PL_2 \rightarrow -$	$s1$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	14	-
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	6	-
		$s3$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	13	-
		NoSt	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	171	-
B3	John: $PL_3 \rightarrow -$	$s1$	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	12	-
	Paul: $PL_5 \rightarrow -$	$s2$	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	5	-
		$s3$	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	11	-
		NoSt	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	167	-
B4	John: $PL_2 \rightarrow -$	$s1$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	14	-
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	6	-
		$s3$	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	13	-
		NoSt	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	171	-
B5	John: $PL_1 \rightarrow -$	$s1$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	6	-
	Paul: $PL_5 \rightarrow -$	$s2$	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	10	-
		$s3$	3 - P: $PL_5(\times)$, $PL_4(\times)$, J: $PL_3(\times)$	10	-
		NoSt	3 - P: $PL_4(\times)$, $PL_5(\times)$, J: $PL_2(\times)$	137	-

7.1.3 Observations from Dialogue Simulations of Test Case C

I present now observations on the results of test case C (Tables 7.5 and 7.6).

- Again strategy $s2$ performs better than strategy $s1$ in terms of number of questions evaluated.
- When Paul starts the dialogue in run C1 Table 7.6, the preferences after the dialogue are: John plan PL_1 and Paul plan PL_5, John selects PL_1 because between these two plans it is the plan that promotes more values.
- In run C5 Table 7.6, plan PL_5 and plan PL_1 are rejected because their side effects are as explained for test case A5. Plan PL_5 proposed by Paul demotes $v_1 = money$ which is John's highest ranked value in run C5.

Table 7.5: Test case C when John starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
C1	John: $PL_1 \to PL_1$	$s1$	2 - J: PL_1(✓), P: PL_4(×)	6	PL_1
	Paul: $PL_4 \to -$	$s2$	2 - J: PL_1(✓), P: PL_4(×)	4	PL_1
		$s3$	2 - J: PL_1(✓), P: PL_4(×)	5	PL_1
		NoSt	2 - J: PL_1(✓), P: PL_4(×)	78	PL_1
C2	John: $PL_2 \to -$	$s1$	3 - J: PL_2(×), PL_6(✓), P: PL_4(×)	8	PL_6
	Paul: $PL_4 \to -$	$s2$	3 - J: PL_2(×), PL_6(✓), P: PL_4(×)	5	PL_6
		$s3$	3 - J: PL_2(×), PL_6(✓), P: PL_4(×)	7	PL_6
		NoSt	3 - J: PL_2(×), PL_6(✓), P: PL_4(×)	106	PL_6
C3	John: $PL_3 \to -$	$s1$	3 - J: PL_3(×), PL_4(×), P: PL_5(✓)	7	PL_5
	Paul: $PL_5 \to PL_5$	$s2$	3 - J: PL_3(×), PL_4(×), P: PL_5(✓)	5	PL_5
		$s3$	3 - J: PL_3(×), PL_4(×), P: PL_5(✓)	6	PL_5
		NoSt	3 - J: PL_3(×), PL_4(×), P: PL_5(✓)	169	PL_5
C4	John: $PL_2 \to -$	$s1$	3 - J: PL_2(×), PL_6(✓), $P : PL_4$(×)	8	PL_6
	Paul: $PL_4 \to -$	$s2$	3 - J: PL_2(×), PL_6(✓), P: PL_4(×)	5	PL_6
		$s3$	J: 3 - PL_2(×), PL_6(✓), P: PL_4(×)	9	PL_6
		NoSt	J: 3 - PL_2(×), PL_6(✓), P: PL_4(×)	106	PL_6
C5	John: $PL_1 \to -$	$s1$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓) P: PL_5(×)	10	PL_6
	Paul: $PL_5 \to -$	$s2$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓) P: PL_5(×)	12	PL_6
		$s3$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓) P: PL_5(×)	11	PL_6
		NoSt	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓) P: PL_5(×)	149	PL_6

TABLE 7.6: Test case C when Paul starts the dialogue

Test case	Preferred plan before → after	St	Proposals	No. Q.	Plan selected
C1	John: $PL_1 \rightarrow PL_1$	$s1$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_1(\checkmark)$	7	PL_1
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_1(\checkmark)$	5	PL_1
		$s3$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_1(\checkmark)$	7	PL_1
		NoSt	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_1(\checkmark)$	138	PL_1
C2	John: $PL_2 \rightarrow -$	$s1$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	8	PL_5
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	5	PL_5
		$s3$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	7	PL_5
		NoSt	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	129	PL_5
C3	John: $PL_3 \rightarrow -$	$s1$	2 - P: $PL_5(\checkmark), J : PL_3(\times)$	6	PL_5
	Paul: $PL_5 \rightarrow PL_5$	$s2$	2 - P: $PL_5(\checkmark), J : PL_3(\times)$	4	PL_5
		$s3$	2 - P: $PL_5(\checkmark), J : PL_3(\times)$	5	PL_5
		NoSt	2 - P: $PL_5(\checkmark), J : PL_3(\times)$	84	PL_5
C4	John: $PL_2 \rightarrow -$	$s1$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	8	PL_5
	Paul: $PL_4 \rightarrow -$	$s2$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	5	PL_5
		$s3$	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	7	PL_5
		NoSt	3 - P: $PL_4(\times), PL_5(\checkmark), J : PL_2(\times)$	126	PL_5
C5	John: $PL_1 \rightarrow -$	$s1$	3 - P: $PL_5(\times), PL_4(\times), J : PL_1(\times)$	7	–
	Paul: $PL_5 \rightarrow -$	$s2$	3 - P: $PL_5(\times), PL_4(\times), J : PL_1(\times)$	9	–
		$s3$	3 - P: $PL_5(\times), PL_4(\times), J : PL_1(\times)$	8	–
		. NoSt	3 - P: $PL_5(\times), PL_4(\times), J : PL_1(\times)$	105	–

7.1.4 Observations from Dialogue Simulations of Test Case D

In test case D strategy $s1$ performs better because most of the inconsistencies are in the initial state. Since suitability questions are considered first (e.g. critical question *CQPP-05: Does the new circumstances already pertain?*) the dialogue ends faster. The results of test case D are presented in Tables 7.7 and 7.8.

Table 7.7: Test case D when John starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
D1	John: $PL_1 \rightarrow PL_4$	$s1$	5 - J: PL_1, PL_2, PL_3,PL_6(×), P: PL_4(✓)	13	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	5 - J: PL_1, PL_2, PL_3,PL_6(×), P: PL_4(✓)	15	PL_4
		$s3$	5 - J: PL_1, PL_2, PL_3,PL_6(×), P: PL_4(✓)	13	PL_4
		NoSt	5 - J: PL_1, PL_2, PL_3,PL_6(×), P: PL_4(✓)	205	PL_4
D2	John: $PL_2 \rightarrow PL_4$	$s1$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	13	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	15	PL_4
		$s3$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	13	PL_4
		NoSt	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	182	PL_4
D3	John: $PL_3 \rightarrow PL_5$	$s1$	5 - J: PL_3, PL_6, PL_2,PL_1(×), P: PL_5(✓)	13	PL_5
	Paul: $PL_5 \rightarrow PL_5$	$s2$	5 - J: PL_3, PL_6, PL_2,PL_1(×), P: PL_5(✓)	15	PL_5
		$s3$	5 - J: PL_3, PL_6, PL_2,PL_1(×), P: PL_5(✓)	13	PL_5
		NoSt	5 - J: PL_3, PL_6, PL_2,PL_1(×), P: PL_5(✓)	198	PL_5
D4	John: $PL_2 \rightarrow PL_4$	$s1$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	13	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	15	PL_4
		$s3$	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	13	PL_4
		NoSt	5 - J: PL_2, PL_6, PL_1,PL_3(×), P: PL_4(✓)	182	PL_4
D5	John: $PL_1 \rightarrow -$	$s1$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓), P: PL_5(×)	8	–
	Paul: $PL_5 \rightarrow -$	$s2$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓), P: PL_5(×)	12	–
		$s3$	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓), P: PL_5(×)	12	–
		NoSt	5 - J: PL_1, PL_2, PL_3(×), PL_6(✓), P: PL_5(×)	145	–

Table 7.8: Test-case D when Paul starts the dialogue

Test case	**Preferred plan before → after**	**St**	**Proposals**	**No. Q.**	**Plan selected**
D1	John: $PL_1 \rightarrow PL_4$	$s1$	2 - P: PL_4(✓), J: PL_1(×)	10	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	2 - P: PL_4(✓), J: PL_1(×)	12	PL_4
		$s3$	2 - P: PL_4(✓), J: PL_1(×)	10	PL_4
		NoSt	2 - P: PL_4(✓), J: PL_1(×)	120	PL_4
D2	John: $PL_2 \rightarrow PL_4$	$s1$	2 - P: PL_4(✓), J: PL_2(×)	10	PL_4
	Paul: $PL_4 \rightarrow PL_4$	$s2$	2 - P: PL_4(✓), J: PL_2(×)	12	PL_4
		$s3$	2 - P: PL_4(✓), J: PL_2(×)	10	PL_4
		NoSt	2 - P: PL_4(✓), J: PL_2(×)	74	PL_4
D3	John: $PL_3 \rightarrow PL_5$	$s1$	2 - P: PL_5(✓), J: PL_3(×)	10	PL_5
	Paul: $PL_5 \rightarrow PL_5$	$s2$	2 - P: PL_5(✓), J: PL_3(×)	12	PL_5
		$s3$	2 - P: PL_5(✓), J: PL_3(×)	10	PL_5
		NoSt	2 - P: PL_5(✓), J: PL_3(×)	81	PL_5
D4	John: $PL_2 \rightarrow PL_5$	$s1$	2 - P: PL_5(✓), J: PL_3(×)	10	PL_5
	Paul: $PL_4 \rightarrow PL_5$	$s2$	2 - P: PL_5(✓), J: PL_3(×)	12	PL_5
		$s3$	2 - P: PL_5(✓), J: PL_3(×)	10	PL_5
		NoSt	2 - P: PL_5(✓), J: PL_3(×)	182	PL_5
D5	John: $PL_1 \rightarrow PL_4$	$s1$	3 - P: PL_5(×),PL_4(✓), J: PL_1(×)	13	PL_4
	Paul: $PL_5 \rightarrow PL_4$	$s2$	3 - P: PL_5(×),PL_4(✓), J: PL_1(×)	17	PL_4
		$s3$	3 - P: PL_5(×),PL_4(✓), J: PL_1(×)	17	PL_4
		NoSt	3 - P: PL_5(×),PL_4(✓), J: PL_1(×)	146	PL_4

7.2 Summary of Results

The results of the simulated dialogue runs show that the use of a strategy to select questions within the *PDGP* protocol is beneficial in terms of dialogue coherence. I implemented the dialogue simulations in a scenario where participants have several plans to pose and evaluated the dialogue according to the number of questions required to end the dialogue.

In general any strategy performs better than a random question selection, but the best strategy requires consideration of the problem. A random ordering is as good as an inappropriate order but not as good as the appropriate one. This means the priority order for the questions is important but problem dependent. The general conclusions of the experiments are the following:

- The *PDGP* protocol can handle multiple proposals for different agents but the way the dialogue examples were designed does not have the flexibility desired to allow a fair dialogue independent of the agent that starts the dialogue. The problem could be addressed by re-starting the dialogue with the other agent.
- When an agent prefers a plan it tries to put it forward first and that accelerates the process of accepting or rejecting it whichever is the case.
- The strategy where possibility questions are put first in the questioning order performs better for most cases because inconsistencies are found mainly in the plan representation.
- When posing random questions, in the worst case the respondent agent has to go through all the questions, which is not desirable. Although for automated dialogues the number of questions may not be relevant I believe agents should adhere to a strategy like the one presented to benefit the coherence and the efficiency of the dialogue. It is also beneficial in terms of tracking reasons to explain the results of the dialogue.
- I believe that the order of the questions in the strategy could be tailored for a particular scenario with information from previous dialogues, and furthermore, the strategy could be modified as the dialogue develops to reduce the exchange of information.

The test cases were designed with inconsistencies in the agents beliefs at several levels and the experiments confirmed the correct questions were presented. The characteristics that indicate a preference for a particular question prioritisation can be summarised as follows:

Strategy that considers validity questions first.

- Inconsistencies in the agents' ontology (questions about the validity of elements):

  - Inconsistencies in the action specification.
  - Inconsistencies in values recognised by the agents.

Strategy that considers suitability questions first.

  - Domains where information changes fast (questions about the initial circumstances)
  - Domains where the values of the agents are not the same.

Strategy that considers possibility questions first.

  - Scenarios where agents have different representations of the world (questions about the possibility of elements).

Strategy that considers possibility in time questions first.

  - Domains where the scheduling of actions could be a problem (questions about the scheduling of the actions).

Strategy that considers alternative questions first.

  - Scenarios with reliable plans (alternate plans).

The prioritization of questions can be more precise and put specific questions first depending on the information available to agents regarding the previous points.

7.3 Chapter Summary

The results of the implementation described in the previous chapter have been presented and analysed. The implementation of the two-step strategy shows that the number of questions decreases considerably when compared to the random approach in all the dialogue simulations.

To summarise, the approach on the strategies: the alignment of belief relies on the identification of relevant critical questions about the validity of elements but the resolution of these inconsistencies (introduced by questions) is influenced by the order in which validity questions are put forward in the dialogue. The order in which questions are put forward in a dialogue does make a difference as shown in the strategies comparison. Furthermore, strategies to select critical questions reduce to some extent the overhead in communication and this could help in some multi-agent environments where communication is more costly than internal computation.

The approach to plan selection implemented provides a way for agents to cooperate, while allowing the agents to reach agreements using individual preferences in the dialogue.

The implemented agents performed better when using a strategy in a complex dialogue. The number of questions exchanged between agents is far smaller than using a random approach. The strategy where possibility questions are put first in the questioning order performs better for most cases because inconsistencies are normally found mainly in the plan representation. The use of a strategy when selecting a question is beneficial in all the simulations, although different strategies performed better in different cases. Furthermore, the characteristics which influence the performance of the strategies were identified. The main differences between this approach and standard distributed planning approaches is *when* and *how* agents discuss the best course of action to take.

The dialogue can then focus on the evaluation of plans considering agents' potentially different beliefs about the world and different preferences. The preferences are applied in the dialogue, to choose the best possible plan taking into account the preferences of both agents. I believe this approach presents an advantage over distributed planning approaches where knowledge-bases are first merged, then plans are created, and there is no clear indication as to where and when agents apply preferences over actions or plans. The detailed critique methodology presented enables such intricate, but important details to be addressed precisely.

Chapter 8

Conclusions

With this chapter I conclude the thesis presenting a summary of the contributions in Section 8.1. In Section 8.2, I present an evaluation of the thesis discussing its strengths and limitations. Finally, Section 8.3 discusses areas of possible future research work.

8.1 Summary of Contributions

The research questions presented in the introduction chapter of this thesis inquire about the necessary elements that autonomous agents need to have in order to propose and select a plan in distributed planning scenarios. Multi-agent planning is known to be a highly complex and detailed problem because of the need to represent and reason about a large number of elements. The approach presented in this thesis considers the process of agreeing on a plan with the use of a dialogue game where agents attack and defend plan proposals. Specifically, the research presented contributes to the solution of problems related to multi-agent planning communication where agents need to agree on plans given that the agents have possibly different views of the world and possibly different beliefs about other agents' capabilities. The contributions (related to the research questions) are:

- **Research Question**: What planning elements or features are relevant when agents discuss multi-agent temporal plans?
- **Research Question**: What argumentation scheme is appropriate to allow agents engage in a dialogue about multi-agent plans?
- **Research Question**: Which critical questions match the argumentation scheme for plan proposals to argue about plans?

  Contribution: *A novel argumentation scheme for plan proposals ASP and a novel list of critical questions related to the scheme.*

  The importance of this argumentation scheme is that it enables agents to propose a plan proposal and allows other agents to critique the proposal rationally. The scheme allows the duration and relative timing of the action to be considered.

Furthermore, a novel list of critical questions related to the argumentation scheme has been defined considering:

- The plan as a whole.
- The individual actions for each agent.
- The different ways in which two actions can be combined.
- The alternative options to both actions and plans.
- The side effects of the actions and plans in relation to the values they promote.

The critical questions enable the proposed plan to be questioned and/or challenged in a comprehensive and structured way, allowing the plan to be defended and refined by the proponent. The critical questions address all the elements of a proposed plan and so they are comprehensive with respect to the chosen representation for plan proposals. Every component and every interaction of components in the plan proposal representation is subject to possible questioning. These questions can identify the specific points at which the plan needs to be refined.

- **Research Question**: What type of protocol would be adequate to allow agents to engage in a dialogue about the critique and modification of multi-agent plans?

  Contribution: *A dialogue game protocol that enables multi agent plan proposals and their evaluation.*

  Following the definition of the argumentation scheme for plan proposals and the related critical questions I defined the *PDGP* dialogue game protocol. The importance of this protocol is that it allows agents to argue about a plan in a dialogue allowing both persuasion and deliberation and to consider multiple proposals. I presented the *syntax* and *semantics* of the *PDGP* protocol that enables plan proposals to be questioned and/or attacked under certain rules.

  The protocol combines locutions from two types of argumentation dialogues to enable agents to persuade and deliberate in the context of a plan evaluation. The protocol is combination of elements from persuasion dialogue game protocols in [10, 24] and deliberation protocols in [106] for multi-agent systems. With this protocol agents are able to engage a dialogue to agree on a plan to execute under a persuasion dialogue and in addition, agents can, when necessary, deliberate on a course of action within the same protocol.

  Whether the proposed plan of action survives such questions and attacks in the dialogue will depend upon the facts about the world underlying the proposal and the ability of the proponent agent to defend his proposal from attack. Consequently, the acceptability or otherwise of the proposed plan will depend upon the outcome of the multi-agent dialogue based upon the critical questions, and vice-versa.

- **Research Question**: How can agents identify, prioritise, and choose a relevant critical question in a dialogue about plans?

Contribution *A methodology to identify, prioritise and select critical questions in a dialogue.*

The methodology to identify, prioritise and select critical questions has two main stages: (1) the identification of relevant questions according to the information presented in the proposal, and (2) a prioritisation criteria according to the question type. The identification of relevant questions is done analysing the elements presented in the proposal and validating them against the local beliefs of the respondent agent plus questions that relate to side effects and alternative options. The resultant "priority orderings" are based on the identification of the nature of the questions in order to present a relevant question in the dialogue. Critical questions are prioritised taking into account factors that could represent points of disagreement in specific scenarios. This prioritisation aims to benefit the dialogue in terms of efficiency (number of questions exchanged) and ultimately provide a structured way to select a plan to execute. I have empirically investigated the orderings and the Although different prioritisation work better in different scenarios, I provided characteristics of the situations which favour different orderings. The strategies help then to improve the length of the dialogue and avoid the exchange of redundant information that could be a significant communication overhead.

- **Research Question**: How effective are the different strategies for choosing critical questions?

  Contribution: *A computational implementation of the argumentation scheme and critical questions, the dialogue protocol, the strategy to select critical questions and a simulated dialogue between two agents, which has produced results which enable to assess the effectiveness of the protocol and the strategies.*

  The implementation simulates a dialogue between two agents that use their preferences over values to propose plans and accept or reject them with the use of critical questions in a dialogue game. The *PDGP* protocol was implemented using tuple centres provided by the *TuCsoN* blackboard architecture where the syntax and the semantics of the protocol were implemented. The semantics of the locutions were defined with the use of the *ReSpecT* language, a tuple language based on first-order logic. Agents use the argumentation scheme for plan proposals formalism from Chapter 3 to post the plan proposals in the tuple centre. The agents were implemented in Java and the dialogue simulations were implemented with the use of the tuple centre as a dialogue container.

  The dialogue focused then on the evaluation of plans considering agents' potentially different beliefs about the world and different preferences. The preferences are applied in the dialogue to choose the best possible plan for both agents. In addition, agents use strategies to identify and prioritise questions in the dialogue.

Contribution: *An evaluation of the strategy to select questions using simulated dialogue runs.*

The evaluation shows how an argumentation-based approach can capture elements from a dialogue about cooperative plans but at the cost of needing to select from a very large number of moves when critiquing a plan proposal, placing a high premium on an effective strategy for move selection. The use of a strategy when selecting a question in a dialogue regarding plans is beneficial, although different strategies perform better in different cases. The experiments confirm that this is the case and show how even a simple strategy can greatly assist in the move selection process. The approach to plan selection presented in this thesis provides means for agents to cooperate, while allowing the agents to reach agreements that reflect their individual preferences. The results of the dialogue runs confirms the strategy should be tailored to the context in which the dialogue develops and modified as the dialogue develops to enhance the overload in the exchange of information

8.2 Evaluation of Work in the Thesis

8.2.1 Strengths

One of the main differences of this approach with standard distributed planning approaches is when and how agents discuss about the best course of action to take. The approach presented uses a persuasion dialogue to critique plans between agents, assuming the plans are already defined. The dialogue focuses then, on the evaluation of plans considering agents' potential different views of the world and different preferences.

Agents' preferences are applied in the dialogue when putting forward their preferred plans according to their preference value ordering in order to choose the best possible plan for both agents. In some distributed planning approaches knowledge-bases are first are merged and then plans are created there is no distinct stage where and when agents apply preferences over actions or plans. The approach presented in this thesis allows agents to cooperate by giving them the means to reach agreements in a structured dialogue using individual preferences in an argumentative scenario.

Using argumentation schemes and critical questions in multi-agent planning related scenarios is useful to explain the gap between automated planning and the scrutability of planning tasks in multi-agent systems and even helpful to translate planning tasks into natural language given the translation of proposals and questions to formal models. This work allows plans to be examined at a finer level of detail than existing approaches.

8.2.2 Limitations

This section presents some of the limitations of the approach in this thesis.

- The critical questions regarding the ways actions are combined only consider two actions. How to decompose these plans into a number of actions and the issues that arise from the interaction of their components is something not considered in this thesis.
- The dialogue simulations implement only two agents; a multi-agent dialogue with several agents critiquing a plan might bring new insights to the approach.
- The dialogue protocol is formed from a combination of three known argumentation protocols but cannot handle cases where agents need to seek information from each other. An approach using protocol combination as in [153] could help to deal with this issue.
- The kind of beliefs of agents is a very specific one since agents do not have any uncertainty. They can be wrong about their beliefs, but they are determined: for every fact, they either believe it or they not.

8.3 Future Work

This section presents some possible future research paths related to each of the main contributions of the thesis presented in the following subsections.

8.3.1 Argumentation Scheme for Plan Proposals

Firstly, the addition of a more extensive set of planning elements such as those defined in the PDDL 3 [75] would be beneficial for specific planning scenarios. In particular, PDDL 3 include constraints and preferences related to actions as well as outcomes which would fit in the ASP argumentation scheme. These extensions would bring new insights on the way plans are proposed and evaluated. The constraints used in PDDL 3 could match the assumed social constraints presented in the argumentation scheme for plan proposals and the preferences would fit into the value promotion schema defined for the plan proposal.

Secondly, the use of values related to the plan is treated in a rather straightforward manner. It would be interesting to define a clear relation between the values promoted in the actions in relation to the values promoted by the plan overall. In other words, to what extent the promoted values of the actions could change when conceptualised as a plan? For example, if a plan is comprised of three actions where the first two promote value v_1 and the remaining action demotes v_1, could it be said the plan promotes value v_1? Also related to the values in the plan is how the preference of values influences the semantics of defining a successful attack (in terms of side effect questions for example). In the implemented dialogue simulations for example a side effect attack is considered successful if the action demotes the highest ranked value of another agent's preference ordering.

Thirdly, it would be worthwhile to consider the use of other semantics to define winning arguments in the dialogue. For example the use of different semantics such as $EAFs$ to model the preferences of agents or hierarchical argumentation [122] would bring new insights to the approach.

Finally, I used the concept of the “social context” in relation to the model in a very straightforward manner. A deeper analysis on constraints related to the proposal would generate a more specific set of critical questions in relation to the social constraints imposed by the domain.

8.3.2 Planning Dialogue Game Protocol

Firstly, extend the protocol with locutions to engage in negotiation and information-seeking for the planning phase and possibly command locutions for the execution phase. Secondly, in terms of the implementation, deploy a multi-agent dialogue within the same protocol and evaluate the performance of both the strategy and the tuple centre. Finally, implement fully the deliberation and persuasion functionalities that the protocol presents, especially action combination.

8.3.3 Strategies in dialogue

Regarding strategies in dialogue, firstly the most straightforward option is to investigate the use of other strategies that further change the question ordering and a different benchmark to evaluate the results.

Secondly, combine the strategy presented with the approach of [133] where recorded dialogues from a simulated military scenario are analysed and the results of the analysis are used to build software agents that support team performance. The research in [133] useful to determine a strategy for argument selection because it classifies utterances and the types of dialogue involved in a planning scenario where the utterances referring to the problem-solving and execution stages are significantly more relevant. An analysis like the one presented in [133] can lead to predefined strategies for critical question selection in specific scenarios. For example, scenarios where problem-solving issues within deliberation dialogues are predominant, predefined questions can be biased towards refinement questions from the complete set of critical questions.

Thirdly, the evaluation results show that in some of the cases (Test-cases A5, C5, D5) there is no plan selected because the proposals get rejected because of their side effects. This suggests that selecting an appropriate question to counter-attack a proposal based on the other agent’s preference is relevant. In [28] the authors present a dialogue system that allows agents to exchange arguments in order to come to an agreement on how to act using a model of what is important to the recipient agent. Assuming an agent can reason about or engage in a dialogue to ask about the other agent’s preference, the strategies presented could take into account this factor when choosing which question to pose. Thus, selecting an appropriate question to counter-attack a proposal based on the other agent’s preference is relevant in the scenario presented.

Another research path is related to coalition formation and the implications of this for the dialogue. Attempts to use dialogue games based on argumentation have been used to create coalitions in [158]. In terms of defining strategy in dialogues it would be relevant to investigate the implications of the strategy used for the coalition and the strategy used in the dialogue since both strategies have points in common. As can be seen from this list of future research paths, much remains to be done in relation with dialogues and planning for autonomous agents.

8.4 Final Summary

The work presented in this thesis contributes to the multi-agent planning communication research in scenarios where agents need to present and agree on a multi-agent plan to execute. With the model to propose plans, the dialogue protocol and the strategy presented in the thesis agents can propose and argue about plan proposals making use of a detailed model to critique such proposals that cover a wide range of possible flaws of the plan presented. By adapting and extending existing research on practical reasoning for multi-agent systems and argumentation- based dialogues, defining methods and strategies to identify and select questions, and presenting an implementation of the model describes this thesis contributes to research in communication in multi-agent planning scenarios.

Appendix A

NGO Force Example

This appendix presents an example scenario in order to explain how the elements presented in the thesis can be instantiated. The *NGO Force example* was first used in [35] in the context of agents' support for mission planning under policy constraints. The example was also used in [170] to describe a model to integrate agent dialogues and plan execution. The scenario here was modified from the original specification in favour of the thesis.

The situation is the following: there are two agents in a war zone, one representing a Non-Governmental Organisation (NGO) and one representing a Peace Keeping Force (PKF). Both agents have a common goal and need to agree on a plan to execute. Agents can move between fourteen different zones in some of which armed conflicts can arise. If the agents travels through some zones (warning zones) they may provoke a conflict, therefore, they will attempt to avoid such zones. The goal of both agents is to help the villagers in zone 14 which can only be performed by NGO but needs the help of PKF to arrive to the zone. The agents have knowledge about their position and about their actions which are durative and have a range that does not affect the outcome of the action

Each agent is able to produce plans for both agents to achieve the overall goal given the agent's internal state. The agents consider also other agents' actions since a single agent may not produce a plan with only its own actions. Each agent, then, has a set of plans that can achieve the goal and are considered as two sequences of actions (one for each agent), with the start times of each action specified (scheduled actions).

Agents have different beliefs about the world, different preferences over values, and can disagree about the different elements of a particular action, therefore, some of their plans are incompatible. Thus, agents will need to resolve these inconsistencies in order to agree on a plan to execute. The scenario has the following conditions:

A.1 Initial situation and goal

- The environment is finite and deterministic.
- The initial situation is the following:
  - Agent NGO is based at zone 1: $inZone(NGO, 2)$
  - Agent PKF is based at zone 3: $inZone(PKF, 3)$
  - Agent NGO believes agent PKF is based at zone 3: $inZone(PKF, 3)$
  - Agent PKF believes agent NGO is based at zone 2: $inZone(NGO, 2)$
- The first goal is to arrive to zone 6. (I will use this goal to exemplify small plans). Note that in some approaches this would be considered as a sub-goal (since the overall goal is to arrive at zone 14 and help the villagers) but I am not using the concept of sub-goals in this thesis.
- The overall goal is to help the villagers in zone 14 where there exists a humanitarian crisis. Agents then need to agree on a plan to arrive to zone 14.

A.2 Zones

- A *zone* can be the following states: secure when there is no conflict or crisis (*zone(SecureZone)*), in warning when a conflict can arise unexpectedly (*zone(InWarning)*), in conflict when there is a conflict in progress (*zone(InConflict)*) or in humanitarian crisis when a zone needs help after an armed conflict (*zone(InHCrisis)*). In Table A.1 are presented the agents beliefs about the zones.

Table A.1: Agent beliefs about zones

Zone		Agent *PKF*	Agent *NGO*
1	*secureZone*	✓	✓
2	*secureZone*	✓	✓
3	*secureZone*	✓	✓
4	*secureZone*	✓	✓
5	*secureZone*	✓	✓
6	*inWarning*	✓	✓
7		*inConflict*	*inWarning*
8		*inConflict*	*secureZone*
9		*secureZone*	*inWarning*
10		*inConflict*	*secureZone*
11	*secureZone*	✓	✓
12	*secureZone*	✓	✓
13	*secureZone*	✓	✓
14	*inHCrisis*	✓	✓
15		*secureZone*	-

A.3 Routes

- To move between zones agents can *only* travel through defined *routes*. See Figures A.1 and A.2 where zones are represented as circles numbered 1-14 and routes represented as arrows between zones and Table A.2.
- Routes are labelled as *safe* ($safeRoute(X, Y)$) and *warning* route ($warningRoute(X, Y)$).
- If agent NGO wants to travel through a *warning route* it has to be escorted by agent PKF.
- Agent PKF can travel all routes.
- Agent NGO can travel safe routes alone and warning routes escorted.
- Agents have some inconsistencies their beliefs about the routes e.g. NGO believes that there is no route between 6 and 8 and that there is a route between 7 and 8 (as shown in Figure A.2). See Table A.2 for an account on the beliefs of agents about routes.

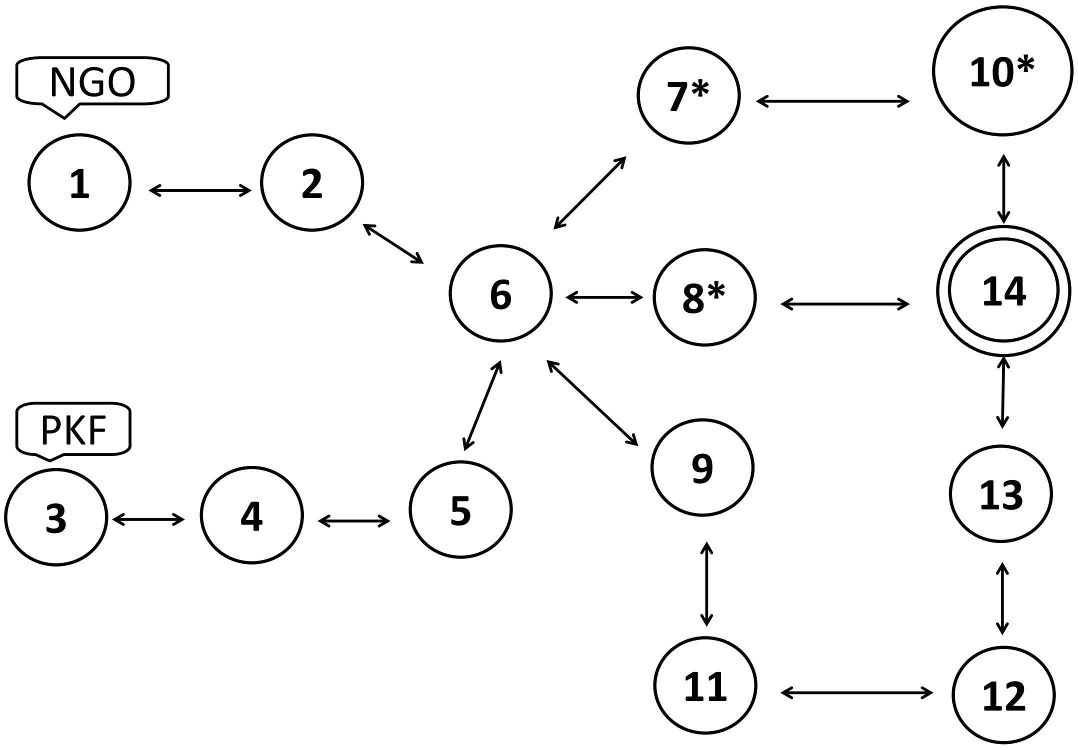


FIGURE A.1: Agent PKF complete view of the zones. A double circle represents the state that needs to be reached. A star (*) represents a conflict.

Table A.2: Agent beliefs about routes

Route		Agent *PKF*	Agent *NGO*
(1,2)	safeRoute	✓	✓
(2,6)	safeRoute		warningRoute
(3,4)	warningRoute	✓	✓
(4,5)	warningRoute	✓	✓
(5,6)	warningRoute	✓	✓
(6,7)	warningRoute	✓	✓
(6,8)		warningRoute	-
(6,9)	warningRoute	✓	✓
(7,8)		warningRoute	-
(7,10)	warningRoute	✓	✓
(8,14)	warningRoute	✓	✓
(9,11)	warningRoute	✓	✓
(11,12)	warningRoute	✓	✓
(10,14)	warningRoute	✓	✓
(12,13)		safeRoute	warningRoute
(13,14)	safeRoute	✓	✓

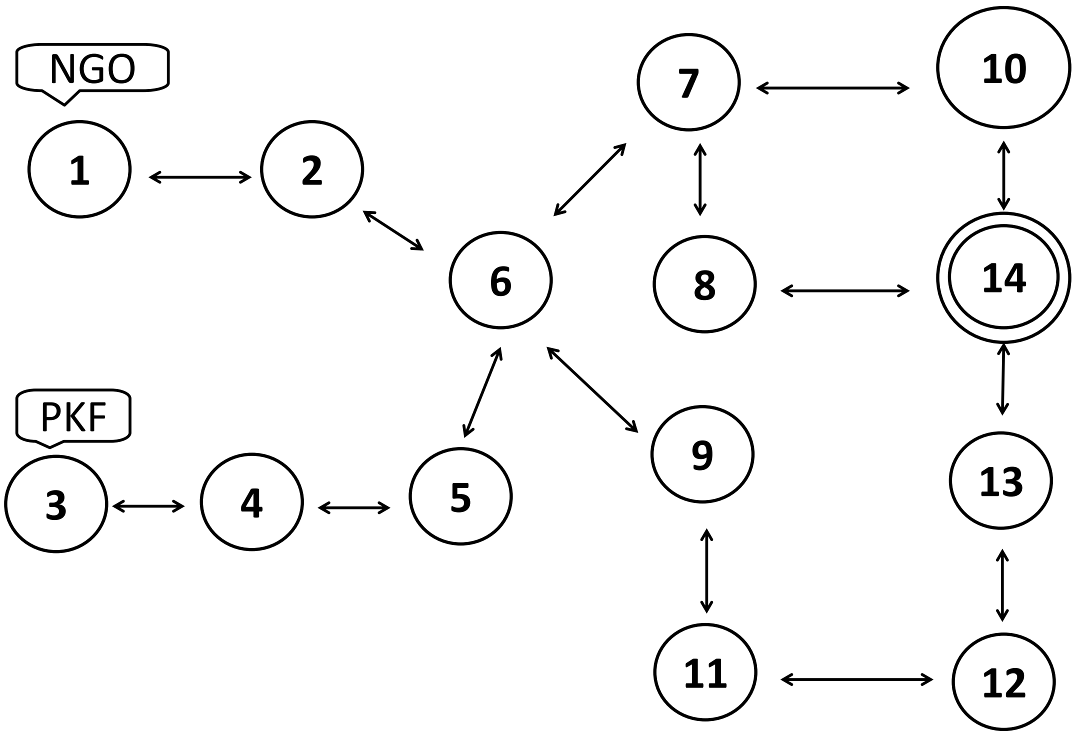


FIGURE A.2: Agent *NGO* complete view of the zones. In contrast with agent *PKF*, *NGO* believes there is no route between zones 6 and 8 and believes that there is a route between zones 7 and 8.

A.4 Actions

The actions that the agents can perform are presented in Tables A.3 and A.4. The agents' beliefs about the other agent actions are presented in Tables **??** and 3.2. Actions have a duration and a range that indicates that the duration of the action can increase or decrease within the range without affecting the action conditions or effects.

TABLE A.3: Actions for Agent PKF

Actions	**Description**	**Duration**	**Range**
$move(PKF, X, Y)$	where X is the initial zone and Y is the end zone and the action results in agent Ag_i in zone Y. (X, Y) must be an existing route.	4	1
$idle(PKF, X)$	where X is the zone where agent Ag_i stays idle.	1	0
$control(PKF, X)$	where X is the zone where agent PKF controls a conflict.	5	2
$protect(PKF, NGO, X)$	where X is the zone where agent PKF controls a conflict and protects agent NGO in zone X.	8	1
$escort(PKF, NGO, X, Y)$	where X is the initial zone and Y is the end zone and agent PKF escort agent NGO when travelling from X to Y ((X, Y) must be an existing route). Note that escort takes more time than move.	4	1
$checkWeapons(PKF)$	where PKF check the state of the weapons	1	0
$askforBackup(PKF)$	where PKF ask for backup during a conflict	1	0

TABLE A.4: Actions for Agent *NGO*

Actions	**Description**	**Duration**	**Range**
$move(NGO, X, Y)$	where X is the initial zone and Y is the end zone and the action results in Agent Ag_i in zone Y. (X, Y) must be an existing route.	4	1
$idle(NGO, X)$	where X is the zone where agent Ag_i stays idle.	1	0
$moveEscorted(NGO, X, Y)$	where X is the initial zone and Y is the end zone and the action results in Agent NGO in zone Y escorted by PKF. (X, Y) must be an existing route.	3	0
$help(NGO, X)$	where X is the zone where agent NGO deploys the humanitarian help.	4	3
$prepareResources(NGO)$	where agent NGO prepares the resources to deploy the humanitarian help.	2	1
$preparePersonnel(NGO)$	where agent NGO prepares the personnel to deploy the humanitarian help.	1	0

A.4.1 Action Elements

Actions are comprised by actions elements or *predicates* used to define the how actions behave in terms of their conditions and effects.

- *fuel*: is a predicate related to each agent and and can be in three states: *Full*, *Decreasing*, *Empty*.
- *groundForce*: represents the soldiers in charge of controlling the conflicts in a zone. The element can be in three states: *Ready*, *Deployed*, *Helping*, *CheckingWeapons* or *Vigilant*
- *aidForce*: represents the volunteers in charge of helping the villagers in a zone in crisis. The element can be in two states: *Ready* or *Deployed*.
- *aidResources*: are the resources used to help the villagers in a zone. The element can be: *Full*, *Decreasing*, *Low* or *Empty*.
- *commChannel*: is the communication channel available to the PKF agent to when escorting. The element can be: *Ready* or*Open*.

A.4.2 Durative Actions Specification

The complete specification of durative actions is presented in Tables A.5 - A.14 (actions with different specifications for each agent are presented using a bold font).

Table A.5: Durative elements for the action *move*() for both agents.

Action	NGO Action elements	PKF Action elements
$move(ag, X, Y)$	**Preconditions** $inZone(ag, X)$ $fuel(Full)/fuel(Low)$ $route(X, Y)$ $secureZone(X)$ $safeRoute(X, Y)$ $\boldsymbol{startTime(beforeNoon)}$ **Start effects** $ag_i(Moving)$ $fuel(Decreasing)$ **Invariant conditions** $fuel(Full)$ **Termination conditions** - **End effects** $inZone(Ag, Y)$ $fuel(Low)/fuel(Empty)$	**Preconditions** $inZone(ag, X)$ $fuel(Full)/fuel(Low)$ $route(X, Y)$ $secureZone(X)$ $safeRoute(X, Y)$ **Start effects** $ag_i(Moving)$ $fuel(Decreasing)$ **Invariant conditions** $fuel(Decreasing)$ **Termination conditions** $\boldsymbol{inZone(Truck, X)}$ **End effects** $inZone(Ag, Y)$ $fuel(Low)/fuel(Empty)$ $\boldsymbol{aidResources(Low)}$

TABLE A.6: Durative elements for the actions $moveEscorted()$

Action	**NGO Action elements**	**PKF Action elements**
$moveEscorted$ $(NGO, PKF,$ $X, Y)$	**Preconditions** $inZone(NGO, X)$ $inZone(PKF, X)$ $fuel(Full)/fuel(Low)$ $\boldsymbol{GroundForce(deployed)}$ $route(X, Y)$ **Start effects** $Ag_i(Moving)$ $fuel(Decreasing)$ **Invariant conditions** $fuel(Decreasing)$ **Termination conditions** **End effects** $inZone(Ag, Y)$ $fuel(Low)/fuel(Empty)$	**Preconditions** $inZone(NGO, X)$ $inZone(PKF, X)$ $fuel(Full)/fuel(Low)$ $\boldsymbol{GroundForce(vigilant)}$ $route(X, Y)$ **Start effects** $Ag_i(Moving)$ $fuel(Decreasing)$ **Invariant conditions** $fuel(Decreasing)$ **Termination conditions** **End effects** $inZone(Ag, Y)$ $fuel(Low)/fuel(Empty)$

Table A.7: Durative elements for the action *escort*() according to both agents.

Action	**NGO Action elements**	**PKF Action elements**
escort (PKF, NGO, X, Y)	**Preconditions** $GroundForce(ready)$ $fuel(Full)/fuel(Low)$ $inZone(PKF, X)$ $inZone(NGO, X)$ $ComChannel(ready)$ $route(X, Y)$ $secureZone(X)$ **Start effects** $\boldsymbol{GroundForce(deployed)}$ $fuel(Decreasing)$ $PKF(Moving)$ $ComChannel(open)$ **Invariant conditions** $ComChannel(open)$ **Termination conditions** - **End effects** $inZone(NGO, Y)$ $inZone(PKF, Y)$ $fuel(Low)/fuel(Empty)$ $\boldsymbol{GroundForce(deployed)}$ $ComChannel(ready)$	**Preconditions** $GroundForce(ready)$ $fuel(Full)/fuel(Low)$ $inZone(PKF, X)$ $inZone(NGO, X)$ $ComChannel(ready)$ $route(X, Y)$ $secureZone(X)$ **Start effects** $\boldsymbol{GroundForce(vigilant)}$ $fuel(Decreasing)$ $PKF(Moving)$ $ComChannel(open)$ $\boldsymbol{flanks(covered)}$ **Invariant conditions** $ComChannel(open)$ **Termination conditions** - **End effects** $inZone(NGO, Y)$ $inZone(PKF, Y)$ $fuel(Low)/fuel(Empty)$ $\boldsymbol{GroundForce(ready)}$ $ComChannel(ready)$

TABLE A.8: Durative elements for the action *idle*()

Action	**Action elements**
$idle(Ag, X)$	**Preconditions** $inZone(Ag, X)$ **Start effects** **Invariant conditions** **Termination conditions** **End effects** $inZone(Ag, X)$

TABLE A.9: Durative elements for the action *control*().

Action	**Action elements**
$control(PKF, X)$	**Preconditions** $GroundForce(ready)$ $inConflict(X)$ $inZone(PKF, X)$ **Start effects** $GroundForce(deployed)$ **Invariant conditions** $GroundForce(deployed)$ **Termination conditions** $GroundForce(ready)$ **End effects** $secureZone(X)$

TABLE A.10: Durative elements for the action *protect*()

Action	**Action elements**
$protect$ (PKF, NGO, X)	**Preconditions** $GroundForce(ready)$ $inZone(NGO, X)$ $inZone(PKF, X)$ $inConflict(X)$ **Start effects** $GroundForce$ $(deployed)$ **Invariant conditions** $GroundForce$ $(deployed)$ **Termination conditions** $secureZone(X)$ **End effects** $GroundForce(ready)$

Table A.11: Durative elements for the action *control*().

Action	**Action elements**
$control(PKF, X)$	**Preconditions** $GroundForce(ready)$ $inConflict(X)$ $inZone(PKF, X)$ **Start effects** $GroundForce(deployed)$ **Invariant conditions** $GroundForce(deployed)$ **Termination conditions** $GroundForce(ready)$ **End effects** $secureZone(X)$

Table A.12: Durative action elements for actions *prepareResources*(), *preparePersonnel*()

Action	**Action elements**	**Action**	**Action elements**
$prepareResources$ (NGO, X)	**Preconditions** $inZone(NGO, X)$ $aidResources(Low)/$ $aidResources(Empty)$ **Start effects** - **Invariant conditions** - **Termination conditions** - **End effects** $aidResources(Full)$	$preparePersonnel$ (NGO, X)	**Preconditions** $inZone(Ag, X)$ AidForce(used) **Start effects** - **Invariant conditions** - **Termination conditions** - **End effects** AidForce(ready)

Table A.13: Durative action elements for the actions *help*()

Action	**NGO Action elements**	**PKF Action elements**
$help(NGO, X)$	**Preconditions** $aidResources(Full)$ $inHCrisisZone(X)$ $inZone(NGO, X)$ $AidForce(ready)$ **Start effects** $AidForce(deployed)$ $aidResources$ $(Decreasing)$ $GroundForce(helping)$ **Invariant conditions** $inZone(NGO, X)$ **Termination conditions** $aidResources(Empty)$ $securedZone(X)$ **End effects** $aidResources(Low)$ $AidForce(used)$	$aidResources(Full)$ $inHCrisisZone(X)$ $inZone(NGO, X)$ $AidForce(ready)$ **Start effects** $AidForce(deployed)$ $aidResources$ $(Decreasing)$ $GroundForce(helping)$ **Invariant conditions** $inZone(NGO, X)$ $\boldsymbol{zone(protected)}$ **Termination conditions** $aidResources(Empty)$ $securedZone(X)$ **End effects** $aidResources(Low)$ $AidForce(used)$

TABLE A.14: Durative action elements for the actions *checkWeapons*

Action	**Action elements**
$checkWeapons(PKF, X)$	**Preconditions** $inZone(PKF, X)$ **Start effects** GroundForce(checkingWeapons) **Invariant conditions** - **Termination conditions** - **End effects** $weapons(Checked)$

A.5 Values

The values associated with the example for both agents are (see Table A.16):

TABLE A.15: Values for the *NGO* example.

Value	*NGO*	*PKF*
v_{NGOsec}	✓	✓
v_{sec}	✓	✓
v_{tt}	✓	✓
v_{mob}	✓	✓
v_{hh}	✓	✓
$v_{saveResources}$	✓	×

– *NGO* Security v_{NGOsec}: Agent *NGO* avoiding zones in conflict.

– Zone security v_{sec}: Travel avoiding warning zones.

– Time of travel v_{tt}: Travel through short routes with a duration no more than 17.

– Mobility v_{mob}: Travel without waiting (idle) in zones.

– Humanitarian help v_{hh}: Help the villagers in a zone in crisis.

Additionally agent *NGO* has the value

The preference ordering of the values for both agents is the following (the order of these values is subjective following the concept of VAFs presented in Section 2.2.2.2):

– *NGO*: $v_{NGOsec} > v_{hh} > v_{tt} > v_{mob} > v_{sec} > v_{saveResources}$

– *PKF*: $v_{sec} > v_{mob} > v_{tt} > v_{NGOsec} > v_{hh}$

TABLE A.16: Description of the values in the *NGO* force example.

Value	**Promoted**	**Demoted**
v_{NGOsec}	when agents travel *through secure zones*	when agents travel *through a conflict zone*
v_{sec}	when agents travel *through secure zones*	when agents travel *through warning or conflict zones*
v_{tt}	if the task is completed in *less* than duration 17	if the task is completed in *more* than duration 17
v_{mob}	if there are *no idle actions* in the plan	if there are idle actions in the plan
v_{hh}	if villagers are helped when in a zone in humanitarian help	if villagers are not helped when in a zone in humanitarian help

A.6 Plans

The plans available for agent NGO are presented in Tables A.17 - A.20 and for agent PKF in Tables A.21 - A.24. In some cases the format of the action $move()$, $moveEscorted()$ and $escort()$ is presented without the duration in favour of clarity.

A.6.1 NGO Agent Plans

TABLE A.17: NGO Plan PL_B to reach zone 6

t	NGO	PKF
0	$move(NGO, 1, 2, d(3))$	$move(PKF, 3, 4, d(3))$
3	$idle(NGO, 2, d(1))$	$idle(PKF, 4, d(1))$
4	$idle(NGO, 2, d(1))$	$move(PKF, 4, 5, d(3))$
7	$idle(NGO, 2, d(1))$	$idle(PKF, 5, d(1))$
8	$move(NGO, 2, 6, d(3))$	$move(PKF, 5, 6, d(3))$
Duration 11 demotes v_{mov} promotes v_{sec} promotes v_{NGOsec} promotes v_{tt}		

Table A.18: *NGO* plan PL_D to reach zone 14.

t	NGO	PKF
0	$idle(NGO, 6, d(1))$	$escort(PKF, NGO, 6, 7)$
1	$moveEscorted(NGO, 6, 7)$	
4	$idle(NGO, 7, d(1))$	$escort(PKF, NGO, 7, 10)$
5	$moveEscorted(NGO, 7, 10)$	
8	$prepareResources(NGO, 10)$	$escort(PKF, NGO, 10, 14)$
9	$moveEscorted(NGO, 10, 14)$	
12	$help(NGO, 14)$	$idle(PKF, 14, d(1))$
Plan duration 16 demotes v_{NGOsec} promotes v_{mov} promotes v_{tt} promotes v_{hh} demotes v_{sec}		

Table A.19: *NGO* Plan PL_F to reach Zone 14

$\boldsymbol{t}$	NGO	PKF
0	$idle(NGO, 6, d(1))$	$escort(PKF, NGO, 6, 7, d(4))$
1	$moveEscorted(NGO, 6, 7, d(3))$	
4	$prepareResources(NGO, 7, d(2))$	$idle(PKF, 7, d(1))$
5		$escort(PKF, NGO, 7, 8, d(4))$
6	$moveEscorted(NGO, 7, 8, d(3))$	
9	$moveEscorted(NGO, 8, 14, d(3))$	$escort(PKF, NGO, 8, 14, d(4))$
14	$help(NGO, 14, d(4))$	$idle(PKF, 14, d(1))$
Duration 16 demotes v_{NGOsec} promotes v_{mov} promotes v_{tt} promotes v_{hh} demotes v_{sec}		

Table A.20: *NGO* Plan PL_{G1} to reach Zone 14

t	NGO	PKF
0	$idle(NGO, 6)$	$escort(PKF, NGO, 6, 9)$
1	$moveEscorted(NGO, 6, 9)$	
4	$idle(NGO, 9)$	$escort(PKF, NGO, 9, 11)$
5	$moveEscorted(NGO, 9, 11)$	
8	$idle(NGO, 11)$	$escort(PKF, NGO, 11, 12)$
9	$moveEscorted(NGO, 11, 12)$	
12	$prepareResources(NGO, 12)$	$idle(PKF, 12)$
13	$move(NGO, 12, 13)$	$idle(PKF, 12)$
15	$move(NGO, 13, 14)$	$idle(PKF, 12)$
18	$help(NGO, 14)$	$idle(PKF, 12)$
Duration 23 promotes v_{NGOsec} promotes v_{mov} demotes v_{tt} demotes v_{sec} promotes v_{hh}		

A.6.2 PKF Plans

Table A.21: *NGO* plan PL_A to reach zone 6.

t	NGO	PKF
0	$idle(NGO, 1, d(1))$	$move(PKF, 3, 4, d(3))$
1	$idle(NGO, 1, d(1))$	
2	$idle(NGO, 1, d(1))$	
3	$move(NGO, 1, 2, d(3))$	$move(PKF, 4, 5, d(3))$
6	$idle(NGO, 2, d(1))$	$idle(NGO, 5, d(1))$
7	$move(NGO, 2, 6, d(3))$	$move(PKF, 5, 6, d(3))$
Duration 10 promotes v_{mov} promotes v_{sec} promotes v_{NGOsec} promotes v_{tt}		

TABLE A.22: PKF plan PL_C to reach zone 6.

t	NGO	PKF
0	$move(NGO, 1, 2, d(3))$	$move(PKF, 3, 4, d(3))$
3	$move(NGO, 2, 6, d(3))$	$idle(PKF, 4, d(1))$
4		$idle(PKF, 4, d(1))$
5		$idle(PKF, 4, d(1))$
6	$idle(NGO, 6, d(1))$	$move(PKF, 4, 5, d(3))$
7	$idle(NGO, 6, d(1))$	
8	$idle(NGO, 6, d(1))$	
9	$idle(NGO, 6, d(1))$	$move(PKF, 5, 6, d(3))$
Duration 12 promotes v_{NGOsec} promotes v_{mov} promotes v_{sec} promotes v_{tt}		

TABLE A.23: PKF Plan PL_E to reach Zone 14

t	NGO	PKF
0	$idle(NGO, 6)$	$escort(PKF, NGO, 6, 8)$
1	$moveEscorted(NGO, 6, 8)$	-
4	$moveEscorted(NGO, 8, 14)$	$escort(PKF, NGO, 8, 14)$
8	$help(NGO, 14)$	$idle(PKF, 14)$
Duration 12 promotes v_{NGOsec} promotes v_{mov} promotes v_{tt} promotes v_{hh}		

Table A.24: PKF Plan PL_G to reach Zone 14

t	NGO	PKF
0	$idle(NGO, 6)$	$escort(PKF, NGO, 6, 9)$
1	$moveEscorted(NGO, 6, 9)$	
4	$idle(NGO, 9)$	$escort(PKF, NGO, 9, 11)$
5	$moveEscorted(NGO, 9, 11)$	
8	$idle(NGO, 11)$	$escort(PKF, NGO, 11, 12)$
9	$moveEscorted(NGO, 11, 12)$	
12	$idle(NGO, 12)$	$escort(PKF, NGO, 12, 13)$
13	$moveEscorted(NGO, 12, 13)$	
16	$prepareResources(NGO, 13)$	$escort(PKF, NGO, 13, 14)$
17	$moveEscorted(NGO, 13, 14)$	
20	$help(NGO, 14)$	$idle(PKF, 14)$
Duration 24 promotes v_{NGOsec} promotes v_{mov} demotes v_{tt} promotes v_{hh} promotes v_{sec}		

Appendix B

PDGP semantics as *ReSpecT* reactions

Table B.1: *ReSpecT* reaction for the semantics of the *enter_dialogue* locution

```
------------------------------------------------------------------------

reaction(out(enter_dialogue(D,A)),
(out_r(checkValid(loc(enter_dialogue(D,A)))))).

reaction(
out_r(checkValid(loc(enter_dialogue(D,A)))),
(
%Preconditions
rd_r(loc(enter_dialogue(_,_))),  %if read tuple loc(enter()) successfull then..
rd_r(dState(open)), % check dialogue state
no_r(dhistory(enter_dialogue(_,A))), %check previous locution by same agent
no_r(dhistory(open_dialogue(_,A))), %check previous locution by same agent
%Posconditions
out_r(participant(D,A)), %participant in
in_r(n_participants(N)),
N1 is N + 1,
out_r(n_participants(N1)),
out_r(dhistory(enter_dialogue(D,A))), %
in_r(checkValid((_))),
in_r(enter_dialogue(_,_))
)).

reaction(out_r(checkValid(loc(enter_dialogue(D,A)))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(enter_dialogue(_,_)))).
------------------------------------------------------------------------
```

TABLE B.2: *ReSpecT* reaction for the semantics of the *open_dialogue* locution

```
-------------------------------------------------------------------------

reaction(out(open_dialogue(D,A)), %If open(d,g) tuple is inserted
(out_r(checkValid(loc(open_dialogue(D,A)))))).


reaction(
out_r(checkValid(loc(open_dialogue(D,A)))),
(
rd_r(loc(open_dialogue(_,_))),
no_r(dhistory(open_dialogue(_,_))),
rd_r(dState(idle)), % check dialogue state
out_r(dhistory(open_dialogue(D,A))), %  insert dHistory tuple
out_r(dState(open)), %change dState tuple "dialogue open"
%PosConditions
out_r(participant(D,A)),
in_r(n_participants(N)),
N1 is N + 1,
out_r(n_participants(N1)),
in_r(dState(idle)), % remove previoUs state
in_r(checkValid(_)),
in_r(open_dialogue(_,_))
)).

%if something fails..just clean
reaction(
out_r(checkValid(loc(open_dialogue(D,A)))),
(in_r(checkValid(_)),
in_r(open_dialogue(_,_)))).

-------------------------------------------------------------------------
```

Table B.3: *ReSpecT* reaction for the semantics of the *propose*() locution

```
----------------------------------------------------------------------

reaction(
out(propose_plan(D,A,P,Id)),
(out_r(checkValid(propose_plan(D,A,P,Id))))).

reaction(
out_r(checkValid(propose_plan(D,A,P,Id))),
(%Preconditions
rd_r(loc(propose_plan(_,_,_,_))),  % read tuple loc(propose()) successfull ..
rd_r(dState(open)),
rd_r(participant(D,A)),
%Posconditions
out_r(dhistory(propose_plan(D,A,P,Id))),
out_r(dState(proposing)),
in_r(dState(open)),
out_r(role(D,proponent,A)), %assign a dialogue role to the agent
in_r(checkValid((_))),
in_r(propose_plan(_,_,_,_)))).

reaction(out_r(checkValid(propose_plan(D,A,P,Id))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(propose_plan(_,_,_,_)))).

----------------------------------------------------------------------
```

Table B.4: *ReSpecT* reaction for the semantics of the *question* locution

```
---------------------------------------------------------------------------

reaction(out(question(D,A,P,Layer,Number,Id,Element,E2,Type)),
(out_r(checkValid(question(D,A,P,Layer,Number,Id,Element,E2,Type))))).

reaction(
out_r(checkValid(question(D,A,P,Layer,Number,Id,Element,E2,Type))),
(rd_r(loc(question(_,_,_,_,_,_,_,_,_))),
rd_r(dState(evaluating)),
rd_r(participant(D,A)),
in_r(n_questions(NQ)),
NQ1 is NQ + 1,
out_r(n_questions(NQ1)),
out_r(dhistory(question(D,A,P,Layer,Number,Id,Element,E2,Type))),
in_r(checkValid((_))),
in_r(question(_,_,_,_,_,_,_,_,_)))).

reaction(out_r(checkValid(question(D,A,P,Layer,Number,Id,Element,E2,Type))),
(in_r(checkValid(_)),
in_r(question(_,_,_,_,_,_,_,_,_)))).

----------------------------------------------------------------------------
```

Table B.5: *ReSpecT* reaction for the semantics of the *assert* locution

```
-----------------------------------------------------------------

reaction(out(assert(D,A,P,Q,E,T)),
(out_r(checkValid(assert(D,A,P,Q,E,T))))).

reaction(
out_r(checkValid(assert(D,A,P,Q,E,T))),
(
rd_r(loc(assert(_,_,_,_,_,_))),
rd_r(dState(evaluating)),
rd_r(participant(D,A)),
rd_r(dhistory(question(D,_,P,_,Q,_,_,_,_))),
out_r(dhistory(assert(D,A,P,Q,E,T))),
in_r(checkValid((_))),
in_r(assert(_,_,_,_,_,_)))).
reaction(out_r(checkValid(assert(D,A,P,Q,E,T))),
(in_r(checkValid(_)),
in_r(assert(_,_,_,_,_,_)))).

-----------------------------------------------------------------
```

Table B.6: *ReSpecT* reaction for the semantics of the *retract* locution

```
--------------------------------------------------------------------------

reaction(out(retract(D,A,P)),
(out_r(checkValid(retract(D,A,P))))). % react -insert check tuple

reaction(
out_r(checkValid(retract(D,A,P))),
(
%Preconditions
rd_r(loc(retract(_,_,_))),  % read tuple loc(propose()) successfull ..
rd_r(dState(evaluating)), % check dialogue state
rd_r(participant(D,A)),
%Posconditions
in_r(dState(evaluating)),
out_r(dState(open)),
out_r(dhistory(retract(D,A,P))), %  insert dHistory tuple
in_r(checkValid((_))), %clean aux tuples
in_r(retract(_,_,_))
)).

reaction(out_r(checkValid(loc(retract(D,A,P)))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(retract(_,_,_))
)).


--------------------------------------------------------------------------
```

Table B.7: *ReSpecT* reaction for the semantics of the *challenge* locution

```
--------------------------------------------------------------------------

reaction(out(challenge(D,A,P,E,C)),
(out_r(checkValid(loc(challenge(D,A,P,E,C)))))). % react -insert check tuple

reaction(
out_r(checkValid(loc(challenge(D,A,P,E,C)))),
(
rd_r(loc(challenge(_,_,_,_,_))),  % read tuple loc(propose()) successfull
rd_r(dState(proposing)), % check dialogue state
rd_r(participant(D,A)),
%Posconditions
in_r(dState(proposing)),
out_r(dState(challenging)),
out_r(dhistory(challenge(D,A,P,E,C))), %  insert dHistory tuple
in_r(checkValid((_))), %clean aux tuples
in_r(challenge(_,_,_,_,_))
)).

reaction(out_r(checkValid(loc(challenge(D,A,P,E,C)))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(challenge(_,_,_,_,_))
)).

--------------------------------------------------------------------------
```

Table B.8: *ReSpecT* reaction for the semantics of the *justify* locution

```
-------------------------------------------------------------------------

reaction(out(justify(D,A,P,E,C)),
(out_r(checkValid(loc(justify(D,A,P,E,C)))))). % react -insert check tuple

reaction(
out_r(checkValid(loc(justify(D,A,P,E,C)))),
(
rd_r(loc(justify(_,_,_,_,_))),  % read tuple loc(propose()) successfull
rd_r(dState(challenging)), % check dialogue state
rd_r(participant(D,A)),
%posconditions
in_r(dState(challenging)),
out_r(dState(proposing)),
out_r(dhistory(justify(D,A,P,E,C))), %  insert dHistory tuple
in_r(checkValid((_))), %clean aux tuples
in_r(justify(_,_,_,_,_))
)).

reaction(out_r(checkValid(loc(justify(D,A,P,E,C)))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(justify(_,_,_,_,_))
)).


-------------------------------------------------------------------------
```

Table B.9: *ReSpecT* reaction for the semantics of the *accept* locution

```
--------------------------------------------------------------------------

reaction(out(accept_proposal(D,A,P)),
(out_r(checkValid(accept_proposal(D,A,P))))). % react -insert check tuple

reaction(
out_r(checkValid(accept_proposal(D,A,P))),
(rd_r(loc(accept_proposal(_,_,_))),  % read tuple loc(propose()) successfull.
rd_r(dState(evaluating)), % check dialogue state
rd_r(participant(D,A)),
no_r(dhistory(accept_proposal(D,A,P))), %check previous locution by same agent
out_r(dhistory(accept_proposal(D,A,P))), %  insert dHistory tuple
out_r(dState(open)),
in_r(checkValid(_)), %clean aux tuples
in_r(accept_proposal(_,_,_))
)).

reaction(out_r(checkValid(accept_proposal(D,A,P))),
(in_r(checkValid(_)),%clean aux tuples
in_r(accept_proposal(_,_,_)))).

--------------------------------------------------------------------------
```

Table B.10: *ReSpecT* reaction for the semantics of the *reject* locution

```
---------------------------------------------------------------------------

reaction(out(reject(D,A,P)),
(out_r(checkValid(loc(reject(D,A)))))). % react -insert check tuple

reaction(
out_r(checkValid(loc(reject(D,A,P)))),
(
rd_r(loc(reject(_,_,_,_))),  % read tuple loc(propose()) successfull ..
rd_r(dState(proposing)), % check dialogue state
rd_r(role(hearer,A)),
rd_r(participant(D,A)),
no_r(dhistory(reject(D,A))), %check previous locution by same agent
out_r(dhistory(rejectept(D,A))), %  insert dHistory tuple
in_r(checkValid((_))), %clean aux tuples
in_r(reject(_,_,_))
)).

reaction(out_r(checkValid(loc(reject(D,A)))),
(
in_r(checkValid(_)),%clean aux tuples
in_r(reject(_,_,_))
)).

---------------------------------------------------------------------------
```

Table B.11: *ReSpecT* reaction for the semantics of the *leave_dialogue* locution

```
--------------------------------------------------------------------

reaction(out(leave_dialogue(D,A)),
(out_r(checkValid(loc(leave_dialogue(D,A)))))).

reaction(
out_r(checkValid(loc(leave_dialogue(D,A)))),
(rd_r(loc(leave_dialogue(_,_))),
rd_r(participant(D,A)),
out_r(dhistory(leave_dialogue(D,A))),
in_r(n_participants(N)),
N1 is N - 1,
out_r(n_participants(N1)),
in_r(checkValid((_))),
in_r(leave_dialogue(_,_))
)).

reaction(out_r(checkValid(loc(leave_dialogue(D,A)))),
(in_r(checkValid(_)),
in_r(leave_dialogue(_,_)))).

--------------------------------------------------------------------
```

Appendix C

Dialogue Run B1, Paul Starts, Strategy $s1$

```
Loading Protocol
[PCADA] Open Dialogue - Agent Proponent
[PCADA] Enter Dialogue - Agent Questioner
[Test] PLAN  NAME: p4 - coach-train
[PCADA] Agent Proponent - Agent 2 - Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
[PCADA] Posting Questions AQ
[PCADA] Question: cQPP04
[PCADA] Question question(d10,7,proposal1427,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal1427,cQPP04,evidence,ValidToken)
[PCADA] Question: cQPP04
[PCADA] Question question(d10,7,proposal1427,4,cQPP04,1324,null,false,
currentCirc_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal1427,cQPP04,evidence,ValidToken)
[PCADA] Question: cQPP05
[PCADA] Question question(d10,7,proposal1427,4,cQPP05,1325,null,true,
currentCirc_POSSIBILITY)
[PCADA] Retract Evidence Proponent
[PCADA] Retract -retract(d10,8,proposal1427)
[PCADA] Retract plan because of question cQPP05
[Test] PLAN  NAME: p5 - train-flight
```

```
[PCADA] Agent Proponent - Agent 2 - Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
[PCADA] Posting Questions AQ
[PCADA] Question: cQPP04
[PCADA] Question question(d10,7,proposal1428,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal1428,cQPP04,evidence,ValidToken)
[PCADA] Question: cQPP04
[PCADA] Question question(d10,7,proposal1428,4,cQPP04,1324,null,false,
currentCirc_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal1428,cQPP04,evidence,ValidToken)
[PCADA] Question: cQPP05
[PCADA] Question question(d10,7,proposal1428,4,cQPP05,1325,null,true,
currentCirc_POSSIBILITY)
[PCADA] Retract Evidence Proponent
[PCADA] Retract -retract(d10,8,proposal1428)
[PCADA] Retract plan because of question cQPP05
[PCADA] Question question(d10,7,proposal1428,6,cQAO01,p1 - coach,null,false,
alternateOption_Plan)
[Test] Alternative PLAN  NAME: p1 - coach
[PCADA] Agent Proponent - Agent 1- Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
[PCADA] Posting Questions AQ
[PCADA] Question: cQPP04
[PCADA] Question question(d10,8,proposal1429,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,7,proposal1429,cQPP04,evidence,ValidToken)
[PCADA] Question: cQPP04
[PCADA] Question question(d10,8,proposal1429,4,cQPP04,1158,null,false,
currentCirc_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,7,proposal1429,cQPP04,evidence,ValidToken)
```

```
[PCADA] Question: cQA04
[PCADA] Question question(d10,8,proposal1429,1,cQA04,1104,1107,true,
precondition_POSS)
[PCADA] Retract Evidence Proponent
[PCADA] Retract -retract(d10,7,proposal1429)
[PCADA] Retract plan because of question cQA04
------------------------------------
Total number of proposals: 3
Total number of questions: 10
------------------------------------
Outcome
------------------------------------
[PCADA] Leave Dialogue Agent Questioner
[PCADA] Leave Dialogue  Agent Proponent
[------Dialogue Finished-----RunB1-Ag2-S1-----
[-----------------------------------------------------------------
```

Appendix D

Dialogue Run C4, John starts, Strategy $s1$

```
Loading Protocol
[PCADA] Open Dialogue - Agent Proponent
[PCADA] Enter Dialogue - Agent Questioner
[Test] PLAN  NAME: p2 - train
[PCADA] Agent Proponent - Agent 1- Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
[PCADA] Posting Questions AQ
[PCADA] Question: cQPP04
[PCADA] Question question(d10,8,proposal7683,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,7,proposal7683,cQPP04,evidence,ValidToken)
[PCADA] Question: cQA04
[PCADA] Question question(d10,8,proposal7683,1,cQA04,7313,7316,true,
precondition_POSS)
[PCADA] Retract Evidence Proponent
[PCADA] Retract -retract(d10,7,proposal7683)
[PCADA] Retract plan because of question cQA04
[Test] PLAN  NAME: p6 - coach-train-flight
[PCADA] Agent Proponent - Agent 1- Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
```

```
[PCADA] Posting Questions AQ
[PCADA] Question: cQPP04
[PCADA] Question question(d10,8,proposal7684,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,7,proposal7684,cQPP04,evidence,ValidToken)
[PCADA] Acccept Proposal AQ
[PCADA] Question question(d10,8,proposal7684,6,cQAO01,
p4 - coach-train,null,false,alternateOption_Plan)
[Test] Alternative PLAN  NAME: p4 - coach-train
[PCADA] Agent Proponent - Agent 2 - Set Active Plan
[PCADA] Agent Proponent Create Proposal Tuple
[PCADA] Post Proposal Tuples AP
[PCADA] Identify  Questions AQ
[PCADA] Applying Ordering Strategy
[PCADA] Posting Questions AQ
[PCADA] Question: cQA01
[PCADA] Question question(d10,7,proposal7685,1,cQA01,7513,
takeCoach_IMM,false,action_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal7685,cQA01,evidence,ValidToken)
[PCADA] Question: cQPP04
[PCADA] Question question(d10,7,proposal7685,1,cQPP04,0,null,false,
norm_VALIDITY)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal7685,cQPP04,evidence,ValidToken)
[PCADA] Question: cQAT01
[PCADA] Question question(d10,7,proposal7685,2,cQAT01,7535,takeTrain_LP,false,
action_POSSIBILITY_TIME)
[PCADA] Provide Evidence Proponent
[PCADA] Assert assert(d10,8,proposal7685,cQAT01,evidence,ValidToken)
[PCADA] Question: cQA12
[PCADA] Question question(d10,7,proposal7685,1,cQA12,7524,7533,true,
endEffect_POSSIBILITY)
[PCADA] Retract Evidence Proponent
[PCADA] Retract -retract(d10,8,proposal7685)
[PCADA] Retract plan because of question cQA12
------------------------------------
Total number of proposals: 3
Total number of questions: 8
------------------------------------
```

```
Outcome Agent -7 p6 - coach-train-flight
------------------------------------
[PCADA] Leave Dialogue Agent Questioner
[PCADA] Leave Dialogue  Agent Proponent
[------Dialogue Finished-----RunC4-Ag1-S1-----
[------------------------------------------------------------------
```

Bibliography

[1] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[2] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, and D. R. Traum. The TRAINS Project: a case study in building a conversational planning agent. *Journal of Experimental and Theoretical Artificial Intelligence*, 7:7–48, 1995.

[3] R. Alur, T. Henzinger, and O. Kupferman. Alternating-time temporal logic. In Willem-Paul de Roever, Hans Langmaack, and Amir Pnueli, editors, *Compositionality: The Significant Difference*, volume 1536 of *Lecture Notes in Computer Science*, pages 23–60. Springer Berlin / Heidelberg, 1998.

[4] L. Amgoud and C. Cayrol. On the acceptability of arguments in preference-based argumentation. In *Proceedings of the Fourteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 1–7, San Francisco, CA, 1998. Morgan Kaufmann.

[5] L. Amgoud and N. Hameurlain. An argumentation-based approach for dialog move selection. In Nicolas Maudet, Simon Parsons, and Iyad Rahwan, editors, *Argumentation in Multi-Agent Systems*, volume 4766 of *Lecture Notes in Computer Science*, pages 128–141. Springer Berlin / Heidelberg, 2007.

[6] L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In *ICMAS '00: Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-2000)*, page 31, Washington, DC, USA, 2000. IEEE Computer Society.

[7] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue and negotiation. *Journal of Artificial Intelligence Research*, 23:2005, 2000.

[8] K. Atkinson and T. Bench-Capon. Action-based alternating transition systems for arguments about action. In *AAAI'07: Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 24–29. AAAI Press, 2007.

[9] K. Atkinson and T. Bench-Capon. Practical reasoning as presumptive argumentation using action based alternating transition systems. *Artificial Intelligence*, 171(10-15):855–874, 2007.

[10] K. Atkinson, T. Bench-Capon, and P. McBurney. A dialogue game protocol for multi-agent argument over proposals for action. In *Proceedings of the First international Conference on Argumentation in Multi-Agent Systems*, ArgMAS'04, pages 149–161, Berlin, Heidelberg, 2005. Springer-Verlag.

[11] K. Atkinson, T. Bench-Capon, and P. McBurney. Computational representation of practical argument. *Synthese*, 152(2):157–206, 2006.

[12] K. Atkinson, T. Bench-Capon, and D. Walton. Distinctive features of persuasion and deliberation dialogues. *Argument and Computation. Accepted for publication. In press*, 2012.

[13] K. Atkinson, R. Girle, P. McBurney, and S. Parsons. Command dialogues. In I. Rahwan and P. Moraitis, editors, *Argumentation in Multi-Agent Systems*, Fifth International Workshop, pages 93–106, Berlin, Heidelberg, 2009. Springer-Verlag.

[14] J. L. Austin. *How to do things with words.* Oxford University Press, Oxford England, 1962.

[15] P. Baroni and Massimiliano Giacomin. Semantics of abstract argument systems. In G. Simari and I. Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 25–44. Springer US, 2009.

[16] A. Belesiotis, M. Rovatsos, and I. Rahwan. A generative dialogue system for arguing about plans in situation calculus. In P. McBurney, I. Rahwan, S. Parsons, and N. Maudet, editors, *Argumentation in Multi-Agent Systems*, volume 6057 of *Lecture Notes in Computer Science*, pages 23–41. Springer Berlin / Heidelberg, 2010.

[17] F. Bellifemine, F. Bergenti, G. Caire, and A. Poggi. Jade: A Java Agent Development Framework. In R. Bordini, M. Dastani, J. Dix, A. Fallah S., and G. Weiss, editors, *Multi-Agent Programming*, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 125–147. Springer US, 2005.

[18] T. Bench-Capon. Specification and implementation of toulmin dialogue game. In *Proceedings of The Foundation for Legal Knowledge Based Systems Conference (JURIX) 98*, pages 5–20. GNI, 1998.

[19] T. J. M. Bench-Capon. Agreeing to differ: modelling persuasive dialogue between parties with different values. *Informal Logic*, 22:2002, 2003.

[20] T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[21] T. J. M. Bench-Capon, S. Doutre, and P. E. Dunne. Audiences in argumentation frameworks. *Artificial Intelligence*, 171(1):42–71, January 2007.

[22] T. J. M. Bench-Capon and H. Prakken. Justifying actions by accruing arguments. In *Proceedings of the 2006 conference on Computational Models of Argument: Proceedings of COMMA 2006*, pages 247–258, Amsterdam, The Netherlands, The Netherlands, 2006. IOS Press.

[23] T.J.M. Bench-Capon and P. E. Dunne. Argumentation in Artificial Intelligence. *Artificial Intelligence*, 171(10-15):619 – 641, 2007.

[24] J. Bentahar, B. Moulin, and B. Chaib-draa. A persuasion dialogue game based on commitments and arguments. In *Proceedings of the International Workshop on Argumentation in Multi-Agent Systems*, pages 148–164, 2004.

[25] J. Bentahar, B. Moulin, J.J. Ch. Meyer, and B. Chaib-draa. A computational model for conversation policies for agent communication. In *Proceedings of the 5th international conference on Computational Logic in Multi-Agent Systems*, CLIMA'04, pages 178–195, Berlin, Heidelberg, 2005. Springer-Verlag.

[26] P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128(12):203 – 235, 2001.

[27] Philippe Besnard and Anthony Hunter. *Elements of Argumentation.* The MIT Press, 2008.

[28] E. Black and K. Atkinson. Choosing persuasive arguments for action. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '11, pages 905–912, Richland, SC, 2011. International Foundation for Autonomous Agents and Multiagent Systems.

[29] E. Black and A. Hunter. An inquiry dialogue system. *Autonomous Agents and Multi-Agent Systems*, 19(2):173–209, October 2009.

[30] A. Bondarenko, F. Toni, and R. A. Kowalski. An assumption-based framework for non-monotonic reasoning. In *Proceedings of the 2nd International Workshop on Logic Programming and Non-monotonic Reasoning*, pages 171–189. MIT Press, 1993.

[31] R. H. Bordini, M. Wooldridge, and J. F. Hübner. *Programming Multi-Agent Systems in AgentSpeak using Jason (Wiley Series in Agent Technology).* John Wiley & Sons, 2007.

[32] C. Boutilier and R. I. Brafman. Planning with concurrent interacting actions. In *In Proceedings of the American Association of Artificial Intelligence (AAAI-97*, pages 720–726, 1997.

[33] M. E. Bratman. Practical reasoning and acceptance in a context. *Mind*, 101(401):1–16, 1992.

[34] M. Brenner. Multiagent planning with partially ordered temporal plans. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1513–1514, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[35] C. Burnett, D. Masato, M. Mccallum, T. J. Norman, J. Giampapa, M. J. Kollingbaum, and K. Sycara. Agent support for mission planning under policy constraints. In *In Proceedings of the Second Annual Conference of the International Technology Alliance, Imperial College*, 2008.

[36] M. W. A. Caminada, W. A. Carnielli, and P. E. Dunne. Semi-stable semantics. *Journal of Logic and Computation*, 2011.

[37] S. Cammarata, D. McArthur, and R. Steeb. Strategies of cooperation in distributed problem solving. In *Proceedings of the Eighth international joint conference on Artificial intelligence - Volume 2*, IJCAI'83, pages 767–770, San Francisco, CA, USA, 1983. Morgan Kaufmann Publishers Inc.

[38] L. Carlson. *Dialogue Games: an Approach to Discourse Analysis.* Reidel Publishind Company, Dordecht, 1983.

[39] D. Cartwright, K. Atkinson, and T. Bench-Capon. Supporting argument in e-democracy. In A Prosser and P. Parycek, editors, *Proceedings of the Third conference on Electronic Democracy (EDEM 2009)*, pages 15–160, Vienna Austria, 2009.

[40] B. Chaib-draa and F. Dignum. Trends in agent communication language. *Computational Intelligence*, 18:89–101, 2002.

[41] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32(3):333–377, July 1987.

[42] C. I. Chesñevar, A. G. Maguitman, and R. P. Loui. Logical models of argument. *ACM Computing Surveys*, 32(4):337–383, December 2000.

[43] A. Cimatti, E. Giunchiglia, F. Giunchiglia, and P. Traverso. Planning via Model Checking: A decision procedure for AR. In *Proceedings of European Conference on Planning (ECP-97)*, pages 130–142. Springer-Verlag, 1997.

[44] P. R. Cohen and H. J. Levesque. Rational interaction as the basis for communication. In P. R. Cohen, J. Morgan, and M. E. Pollack, editors, *Intentions in Communication*, pages 221–256. MIT Press, 1990.

[45] A. Coles, M. Fox, K. Halsey, D. Long, and A. Smith. Managing concurrency in temporal planning using planner-scheduler interaction. *Artificial Intelligence*, 173(1):1 – 44, 2009.

[46] Daniel D. Corkill. Hierarchical planning in a distributed environment. In *Proceedings of the 6th International Joint vonference on Artificial Intelligence*, volume 1 of *IJCAI'79*, pages 168–175, San Francisco, CA, USA, 1979. Morgan Kaufmann Publishers Inc.

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

[48] E. Denti, A. Natali, and A. Omicini. On the Expressive Power of a Language for Programming Coordination Media. In *In Proceedings of the 1998 ACM Symposium on Applied Computing (SAC98)*, pages 169–177, 1998.

[49] M. E. desJardins, E. H. Durfee, C. L. Ortiz, Jr., and M. J. Wolverton. A survey of research in distributed, continual planning. *AI Magazine*, 20(4):13–20, 2000.

[50] J. Devereux and C. Reed. Strategic argumentation in rigorous persuasion dialogue. In *Proceedings of the 6th International Conference on Argumentation in Multi-Agent Systems*, ArgMAS'09, pages 94–113, Berlin, Heidelberg, 2010. Springer-Verlag.

[51] F. Dignum, B. Dunin-Keplicz, and R. Verbrugge. Creating collective intention through dialogue. *Logic Journal of the IGPL*, 9(2):305–319, 2001.

[52] Minh Binh Do and Subbarao Kambhampati. Sapa: A multi-objective metric temporal planner. *Journal of Artificial Intelligence Research (JAIR)*, 20:155–194, 2003.

[53] S Doutre, P. McBurney, M. Wooldridge, and W. Barden. Information-seeking agent dialogs with permissions and arguments. Technical Report ULCS-05-010, Department of Computer Science. University of Liverpool, Liverpool UIK, 2005.

[54] S. Doutre and J. Mengin. Preferred extensions of argumentation frameworks: Query, answering, and computation. In R. Gor, A. Leitsch, and T. Nipkow, editors, *Automated Reasoning*, volume 2083 of *Lecture Notes in Computer Science*, pages 272–288. Springer Berlin / Heidelberg, 2001.

[55] T. B. Downing. *Java RMI: Remote Method Invocation.* IDG Books Worldwide, Inc., Foster City, CA, USA, 1st edition, 1998.

[56] P. M. Dung. On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[57] P.M. Dung, R. Kowalski, and F. Toni. Assumption-based argumentation. In G. Simari and I. Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 199–218. Springer US, 2009.

[58] Barbara Dunin-Keplicz and Rineke Verbrugge. Dialogue in teamwork. In Ricardo Jardim-Gonalves, Jianzhong Cha, and Adolfo Steiger-Garo, editors, *Enhanced Interoperable Systems. Proceedings of the 10th ISPE International Conference on Concurrent Engineering (ISPE CE 2003), July 26-30, 2003, Madeira, Portugal*, pages 121–128. A. A. Balkema Publishers, 2003.

[59] P. E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artif. Intell.*, 171(10-15):701–729, July 2007.

[60] P. E. Dunne. The computational complexity of ideal semantics i: Abstract argumentation frameworks. In *Proceedings of the 2008 conference on Computational Models of Argument: Proceedings of COMMA 2008*, pages 147–158, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.

[61] P. E. Dunne and T. Bench-Capon. Two party immediate response disputes: properties and efficiency. *Artificial Intelligence*, 149(2):221–250, 2003.

[62] E. Durfee and V. R. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, 21:1167–1183, 1991.

[63] E. H. Durfee. Distributed problem solving and planning. In J. G. Carbonell and Jöorg Siekmann, editors, *Mutli-agents systems and applications*, pages 118–149. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[64] E. H. Durfee, V. R. Lesser, and D. Corkill. Trends in cooperative distributed problem solving. *IEEE Transactions on Knowledge and Data Engineering*, 1:63–83, 1995.

[65] C. D. Emele, T. J. Norman, and S. Parsons. Argumentation strategies for plan resourcing. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*, volume 3 of *AAMAS '11*, pages 913–920, Richland, SC, 2011. International Foundation for Autonomous Agents and Multiagent Systems.

[66] E. Ephrati and J. S. Rosenschein. Multi-agent planning as the process of merging distributed sub-plans. In *In Proceedings of the Twelfth International Workshop on Distributed Artificial Intelligence (DAI-93*, pages 115–129, 1993.

[67] G. Ferguson and J. F. Allen. Arguing about plans: Plan representation and reasoning for mixed-initiative planning. In *In Proceedings of the Second International Conference on AI Planning Systems (AIPS-94)*, pages 43–48, 1994.

[68] R. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. In *International Joint Conferences on Artificial Intelligence*, pages 608–620, 1971.

[69] T. Finin, Y. Labrou, and J. Mayfield. KQML as an agent communication language. In J. Bradshaw, editor, *Software Agents*, pages 291–316. MIT Press, 1997.

[70] FIPA. Communicative Act Library Specification. Standard SC00037J, IEEE Foundation for Intelligent Physical Agents, 2002.

[71] M. Fox and D. Long. PDDL2.1: An extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research (JAIR)*, 20:61–124, 2003.

[72] E. Freeman, S. Hupfer, and K. Arnold. *JavaSpaces Principles, Patterns, and Practice: Principles, Patterns and Practices*. The Jini Technology Series. Addison-Wesley Longman, June 1999.

[73] H. Geffner. Perspectives on Artificial Intelligence Planning. In *Eighteenth National Conference on Artificial intelligence*, pages 1013–1023, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[74] D. Gelernter and N. Carriero. Coordination languages and their significance. *Communications of the ACM*, 35(2):97–107, 1992.

[75] A. E. Gerevini, P. Haslum, D. Long, A. Saetti, and Y. Dimopoulos. Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners. *Artificial Intelligence*, 173(5-6):619–668, April 2009.

[76] M. Ghallab, C. K. Isi, S. Penberthy, D. E. Smith, Y. Sun, and D. Weld. PDDL - The Planning Domain Definition Language. Technical report, CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998.

[77] T. Gordon, H. Prakken, and D. Walton. The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10-15):875–896, 2007.

[78] M. Greaves, H. Holmback, and J. Bradshaw. What is a conversation policy? In *Issues in Agent Communication*, pages 118–131, London, UK, 2000. Springer-Verlag.

[79] J. Groenendijk and M. Stokhof. *Questions*, chapter Handbook of Logic and Language, pages 1059–1124. Elsevier, 2nd edition, 2010.

[80] D. Gross, J.F. Allen, and D. R. Traum. The TRAINS-91 dialogues. Technical Report TRAINS Technical Note 92-1, University of Rochester, Rochester, NY, 1993.

[81] B. J. Grosz and S. Kraus. The Evolution of SharedPlans. In A. Rao and M. Wooldridge, editors, *Foundations and Theory of Rational Agency*, pages 227–262. Kluwer Academic Publishers, 1998.

[82] J. Habermas and T. McCarthy. *The Theory of Communicative Action, Volume 1: Reason and the Rationalization of Society.* The Theory of Communicative Action. Beacon, 1985.

[83] K. Halsey, D. Long, and M. Fox. Crikey - a temporal planner looking at the integration of scheduling and planning. In *Proceedings of the Workshop on Integration Scheduling Into Planning at 13th International Conference on Automated Planning and Scheduling (ICAPS'03)*, pages 46–52, June 2004.

[84] C. L. Hamblin. *Fallacies.* Richard Clay (The Chaucer press) Ltd., Bungay, Suffolk, 1970.

[85] B. Hayes-Roth. A blackboard architecture for control. *Artificial Intelligence*, 26(3):251 – 321, 1985.

[86] S. Heras, M. Rebollo, and V. Julián. A dialogue game protocol for recommendation in social networks. In Emilio Corchado, Ajith Abraham, and Witold Pedrycz, editors, *Hybrid Artificial Intelligence Systems*, volume 5271 of *Lecture Notes in Computer Science*, pages 515–522. Springer Berlin / Heidelberg, 2008. 10.1007/978-3-540-87656-4_64.

[87] K. V. Hindriks, F. S. de Boer, W. van der Hoek, and J.J. Ch. Meyer. Agent programming with declarative goals. In *Proceedings of the 7th International Workshop on Intelligent Agents VII. Agent Theories Architectures and Languages*, ATAL '00, pages 228–243, London, UK, 2001. Springer-Verlag.

[88] J. Hintikka. *The Game of Language: Studies in Game-Theoretical Semantics and Its Applications*, volume 22 of *Synthese Language Library.* D. Reidel, Dordrecht, The Netherlands, 1983.

[89] J. Hulstijn. *Dialogue Models for Inquiry and Transaction. PhD Thesis.* PhD thesis, Universiteit Twente, Enschede, The Netherlands, 2000.

[90] J. Hulstijn and L. van der Torre. Combining Goal Generation and Planning in an Argumentation Framework. pages 212–218, 2004.

[91] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(03):223–250, 1993.

[92] N. R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 117:277–296, 2000.

[93] S. Kambhampati. Characterizing multi-contributor causal structures for planning. In *Proceedings of the first international conference on Artificial intelligence planning systems*, pages 116–125, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[94] Craig A. Knoblock. Generating parallel execution plans with a partial-order planner. In *In Proceedings of the Second International Conference on AI Planning Systems (AIPS-94)*, pages 98–103, 1994.

[95] E. M. Kok, J.J. Ch. Meyer, H. Prakken, and G. A. W. Vreeswijk. A formal argumentation framework for deliberation dialogues. In *Proceedings of the 7th International Conference on Argumentation in Multi-Agent Systems*, ArgMAS'10, pages 31–48, Berlin, Heidelberg, 2011. Springer-Verlag.

[96] S. Kraus. *Strategic Negotiation in Multiagent Environments.* MIT Press, Cambridge, MA, USA, 2001.

[97] Y. Labrou and T. Finin. A semantics approach for KQML–A general purpose communication language for software agents. In *CIKM '94: Proceedings of the Third International Conference on Information and Knowledge Management*, pages 447–455, New York, NY, USA, 1994. ACM.

[98] Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: The current landscape. *IEEE Intelligent Systems*, 14(2):45–52, 1999.

[99] l. A. Letia and A. Groza. Planning with argumentation schemes in online dispute resolution. In *Intelligent Computer Communication and Processing, 2007 IEEE International Conference on*, pages 17 –24, sept. 2007.

[100] H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, volume Volume 1 of *AAAI'90*, pages 94–99. AAAI Press, 1990.

[101] M. Luck, R. Ashri, and M. D'Inverno. *Agent-Based Software Development.* Artech House, 2004.

[102] M. Luck, P. McBurney, O. Shehory, and S. Willmott. *Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing).* AgentLink, 2005.

[103] J. D. Mackenzie. Question-begging in non-cumulative systems. *Journal of Philosophical Logic*, 8:117–133, 1979. 10.1007/BF00258422.

[104] N. Maudet and B. Chaib-Draa. Commitment-based and dialogue-game-based protocols: new trends in agent communication languages. *Knowledge Engineering Review*, 17(2):157–179, 2002.

[105] P. McBurney. *Rational Interaction.* PhD thesis, University of Liverpool, 2002.

[106] P. McBurney, D. Hitchcock, and S. Parsons. The Eightfold Way of Deliberation Dialogue. *International Journal of Intelligent Systems*, 22(1):95–132, 2007.

[107] P. McBurney and S. Parsons. Chance discovery using dialectical argumentation. In *Proceedings of the Joint JSAI 2001 Workshop on New Frontiers in Artificial Intelligence*, pages 414–424, London, UK, UK, 2001. Springer-Verlag.

[108] P. McBurney and S. Parsons. Games that agents play: A formal framework for dialogues between autonomous agents. *Journal of Logic, Language and Information*, 11(3):315–334, 2001.

[109] P. McBurney and S. Parsons. Representing epistemic uncertainty by means of dialectical argumentation. *Annals of Mathematics and AI*, 32(1-4):125–169, 2002.

[110] P. McBurney and S. Parsons. Locutions for argumentation in agent interaction protocols. In R. M. van Eijk, M-P. Huget, and F. Dignum, editors, *Agent Communication. Revised Proceedings of the International Workshop on Agent Communication (AC2004)*, volume 3396, pages 209–225, New York, NY, USA, July 2004. Berlin, Germany Springer.

[111] P. McBurney and S. Parsons. Dialogue Games for Agent Argumentation. In I. Rahwan and G. Simari, editors, *Argumentation in Artificial Intelligence*, chapter 13, pages 261–280. Springer, Berlin, Germany, 2009.

[112] P. McBurney, S. Parsons, and M. Wooldridge. Desiderata for agent argumentation protocols. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems.*, AAMAS '02, pages 402–409, New York, NY, USA, 2002. ACM.

[113] P. McBurney, R. M. van Eijk, S. Parsons, and L. Amgoud. A dialogue-game protocol for agent purchase negotiations. *Journal of Autonomous Agents and Multi-Agent Systems*, 7:235–273, 2002.

[114] J. Mccarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence*, pages 463–502. Edinburgh University Press, 1969.

[115] D. G. McVitie and L. B. Wilson. The stable marriage problem. *Communications of the Association for Computing Machinery (ACM)*, 14(7):486–490, July 1971.

[116] R. Medellin-Gasque, K. Atkinson, and T. Bench-Capon. An Analys of Critical Questions for Co-operative Plan Proposals. Technical Report ULCS-12-002, Department of Computer Science, University of Liverpool, UK, September 2011.

[117] R. Medellin-Gasque, K. Atkinson, and T. Bench-Capon. Dialogue game protocol for co-operative plan proposals. Technical Report ULCS-12-003, Department of Computer Science, University of Liverpool, UK, March 2012.

[118] R. Medellin-Gasque, K. Atkinson, T. Bench-Capon, and P. McBurney. Strategies for question selection in argumentative dialogues about plans. *Jounal of Argument and Computation*, 4(2):151–179, 2013.

[119] R. Medellin-Gasque, K. Atkinson, and T. J. M. Bench-Capon. Persuasion strategies for argumentation about plans. In *Proceedings of the Fourth International Conference on Computational Models of Argument*, pages 334–341, 2012.

[120] R. Medellin-Gasque, K. Atkinson, P. McBurney, and T. Bench-Capon. Arguments over co-operative plans. In *Theory and Applications of Formal Argumentation. First International Workshop, TAFA 2011*, Lecture Notes in Computer Science (LNCS) 7132, pages 50–66, Barcelona, Spain, July 2011. Springer, Berlin, Germany.

[121] R. Medellin-Gasque, K. Atkinson, P. McBurney, and T. Bench-Capon. Critical Questions for Plan Proposals. Technical Report ULCS-11-003, Department of Computer Science, University of Liverpool, UK, March 2011.

[122] S. Modgil. Hierarchical argumentation. In *Proceedings of the 10th European conference on Logics in Artificial Intelligence*, JELIA'06, pages 319–332, Berlin, Heidelberg, 2006. Springer-Verlag.

[123] S. Modgil. Reasoning about preferences in argumentation frameworks. *Artificial Intelligence*, 173(9-10):901–934, June 2009.

[124] S. Modgil and T. J. M. Bench-Capon. Metalevel argumentation. *Journal of Logic and Computation*, 2010.

[125] Y. Moses and M. Tennenholtz. Artificial social systems. *Computers and AI*, 14:533–562, 1995.

[126] A. Omicini and E. Denti. From tuple spaces to tuple centres. *Science of Computer Programming*, 41(3):277 – 294, 2001.

[127] A. Omicini and F. Zambonelli. Coordination for internet application development. *Autonomous Agents and Multi-Agent Systems*, 2:251–269, September 1999.

[128] E. Onaindía, O. Sapena, and A. Torreño. Cooperative distributed planning through argumentation. In *International Journal of Artificial Intelligence*, volume 4, pages 118–136. CESER Publications, 2010.

[129] N. Oren, T. J. Norman, and A. Preece. Loose lips sink ships: a heuristic for argumentation. In S. Parsons N. Maudet and I. Rahwan, editors, *Proceedings of the 3rd International Workshop on Argumentation in Multi-Agent Systems*, volume 4766 of *Lecture Notes in Computer Science*, pages 121–134. Springer, 2006.

[130] W. Ouerdane, N. Maudet, and A. Tsoukias. Argument schemes and critical questions for decision aiding process. In *Proceedings of the 2008 Conference on Computational Models of Argument*, pages 285–296, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.

[131] S. Parsons and P. McBurney. Argumentation-based communication between agents. In Marc-Philippe Huget, editor, *Communication in Multiagent Systems*, volume 2650 of *Lecture Notes in Computer Science*, pages 164–178. Springer Berlin Heidelberg, 2003.

[132] S. Parsons and P. McBurney. Argumentation-based dialogues for agent coordination. In *Group Decision and Negotiation*, volume 12, pages 415–439. Springer Netherlands, 2003.

[133] S. Parsons, S. Poltrock, H. Bowyer, and Y. Tang. Analysis of a recorded team coordination dialogue. In *The Second Annual Conference of the International Technology Alliance*, Imperial College London, 2008.

[134] S. Parsons, C. Sierra, and N. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8:261–292, 1998.

[135] S. Parsons, M. Wooldridge, and L. Amgoud. Properties and complexity of some formal inter-agent dialogues. *Journal of Logic and Computation*, 13:347–376, 2003.

[136] S. D. Parsons and N. R. Jennings. Neogotiation through argumentation - a preliminary report. In *Second International Conference on Multi-Agent Systems*, pages 267–274, 1996.

[137] R. S. Patil, R. E. Fikes, P. F. Patel-Schneider, D. McKay, T. Finin, T. Gruber, and R. Neches. The DARPA knowledge sharing effort: progress report. In Michael N. Huhns and Munindar P. Singh, editors, *Readings in agents*, pages 243–254. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[138] J. Pitt and A. Mamdani. Some remarks on the semantics of fipa's agent communication language. *Autonomous Agents and Multi-Agent Systems*, 2(4):333–356, November 1999.

[139] M. E. Pollack. The uses of plans. *Artificial Intelligence*, 57:43–68, 1992.

[140] M. E. Pollack and J. F. Horty. There's more to life than making plans: Plan management in dynamic, multi-agent environments. *AI Magazine*, 20:20–4, 1999.

[141] S. Popkorn. *First Steps in Modal Logic.* Cambridge University Press, 1994.

[142] H. Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation*, 15(6):1009–1040, December 2005.

[143] H. Prakken. Formal systems for persuasion dialogue. *Knowledge Engineering Review*, 21(2):163–188, June 2006.

[144] H. Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1(2):93–124, 2010.

[145] H. Prakken, C. Reed, and D. Walton. Argumentation schemes and burden of proof. In *In Workshop Notes of the Fourth Workshop on Computational Models of Natural Argument*, pages 81–86, 2004.

[146] H. Prakken and G. Sartor. A dialectical model of assessing conflicting arguments in legal reasoning. *Artificial Intelligence and Law*, 4:331–368, 1996.

[147] H. Prakken and G. Sartor. Modelling reasoning with precedents in a formal dialogue game. *Artificial Intelligence and Law*, 6:231–287, 1998. 10.1023/A:1008278309945.

[148] van Eijk R. *Programming Languages for Agent Communication.* PhD thesis, Department of Computer Science, University of Utretch, 2000.

[149] I. Rahwan and L. Amgoud. An argumentation based approach for practical reasoning. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '06, pages 347–354, New York, NY, USA, 2006. ACM.

[150] I. Rahwan, L. Sonenberg, N. R. Jennings, and P. McBurney. Stratum: A methodology for designing heuristic agent negotiation strategies. *Applied Artificial Intelligence*, 21(6):489–527, June 2007.

[151] A. S. Rao and M. P. Georgeff. BDI Agents: From Theory to Practice. In *In Proceedings of the First International Conference on Multi-agent Systems (ICMAS-95)*, pages 312–319, 1995.

[152] J. Raz. *Practical reasoning.* Oxford readings in philosophy. Oxford University Press, 1978.

[153] C. Reed. Dialogue Frames in Agent Communication. In *In Proceedings of the Third International Conference on Multi-Agent Systems*, pages 246–253. IEEE Press, 1998.

[154] C. Reed and D. Walton. Towards a formal and implemented model of argumentation schemes in agent communication. *Autonomous Agents and Multi-Agent Systems*, 11(2):173–188, September 2005.

[155] R. Reiter. Natural actions, concurrency and continuous time in the situation calculus. pages 2–13. Morgan Kaufmann, 1996.

[156] A. Ricci and M. Viroli. Coordination artifacts: A unifying abstraction for engineering environment-mediated coordination in mas. *Informatica (Slovenia)*, 29(4):433–444, 2005.

[157] L. Riley, K. Atkinson, T. Payne, and E. Black. An implemented dialogue system for inquiry and persuasion. In *Proceedings of the First International Workshop on the Theory and Applications of Formal Argumentation (TAFA'11)*, pages 67–84, 2011.

[158] L. Riley, K. Atkinson, and T. R. Payne. A dialogue game for coalition structure generation with self-interested agents. In *Proceedings of the Fourth International Conference on Computational Models of Argument*, pages 229–236, 2012.

[159] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[160] J.R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, 1969.

[161] J.R. Searle. *Rationality in Action*. MIT Press, Cambridge, MA, USA, 2001.

[162] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[163] M. P. Singh. Agent communication languages: Rethinking the principles. *Computer*, 31(12):40–47, 1998.

[164] I.A. Smith, P.R. Cohen, J.M. Bradshaw, M. Greaves, and H. Holmback. Designing conversation policies using joint intention theory. In *Proceedings. International Conference on Multi Agent Systems*, pages 269 –276, July 1998.

[165] B. Streumer. *A Companion to the Philosophy of Action*, chapter Practical Reasoning. Blackwell Companions to Philosophy. John Wiley & Sons, 2010.

[166] K. Sycara and G. Sukthankar. Literature review of teamwork models. Technical Report CMU-RI-TR-06-50, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2006.

[167] K. P. Sycara. Argumentation: Planning Other Agents' Plans. In *International Joint Conference on Artificial Intelligence*, pages 517–523, 1989.

[168] M. Tadiou, A. Shimazu, and T. Nakajima. The state of the art in agent communication languages. *Knowledge Information Systems*, 2(3):259–284, 2000.

[169] Y. Tang, T. J. Norman, and S. Parsons. Agent-based dialogues to support plan execution by human teams. *Annual Conference of International Technologgy Alliance*, September 2008.

[170] Y. Tang, T. J. Norman, and S. Parsons. A model for integrating dialogue and the execution of joint plans. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 883–890, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[171] Y. Tang and S. Parsons. Argumentation-based multi-agent dialogues for deliberation. In S. Parsons, N. Maudet, P. Moraitis, and I. Rahwan, editors, *Argumentation in Multi-Agent Systems*, volume 4049 of *Lecture Notes in Computer Science*, pages 229–244. Springer Berlin / Heidelberg, 2006.

[172] P. Tolchinsky, S. Modgil, and U. Corts. Argument schemes and critical questions for heterogeneous agents to argue over the viability of a human organ for transplantation. In *AAAI Spring Symposium: Argumentation for Consumers of Healthcare'06*, pages 105–105, 2006.

[173] A. Toniolo, T. J. Norman, and K. Sycara. On the benefits of argumentation schemes in deliberative dialogue. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '12, pages 1409–1410, Richland, SC, 2012. International Foundation for Autonomous Agents and Multiagent Systems.

[174] A. Toniolo, K. Sycara, and T.J. Norman. Argumentation schemes for policy-driven planning. In *Proceedings of the First International Workshop on the Theory and Applications of Formal Argumentation*, Barcelona, Spain, 2011.

[175] A. Torreño, E. Onainda, and O. Sapena. Reaching a common agreement discourse universe on multi-agent planning. In *5th International Conference on Hybrid Artificial Intelligence Systems (HAIS 2010)*, volume 6077, pages 185–192. Springer, 2010.

[176] S. Toumlin. *The use of Argument.* Cambridge University Press, 1958.

[177] V. Urovi, S. Bromuri, J. McGinnis, K. Stathis, and A. Omicini. Automating Workflows Using Dialetical Argumentation. *IADIS International Journal on Computer Science and Information System*, 3(2):110–125, 2008.

[178] W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: Effectiveness, feasibility, and synthesis. *Synthese*, 156:1–19, 2007.

[179] W. van der Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In *In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002)*, pages 1167–1174. ACM Press, 2002.

[180] F.H. van Eemeren and R. Grootendorst. *A Systematic Theory of Argumentation. The pragma-dialectical approach.* Cambridge University Press, 2004.

[181] B. Verheij. Dialectical argumentation with argumentation schemes: an approach to legal logic. *Artificial Intelligence and Law*, 11(2-3):167–195, 2003.

[182] S. Vinoski. Corba: integrating diverse applications within distributed heterogeneous environments. *Communications Magazine, IEEE*, 35(2):46 –55, 1997.

[183] G. A. W. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(12):225 – 279, 1997.

[184] G. A. W. Vreeswijk and H. Prakken. Credulous and sceptical argument games for preferred semantics. In *Proceedings of the 7th European Workshop on Logic for Artificial Intelligence (JELIA2000)*, pages 224–238. Springer LNAI, 2000.

[185] D. N. Walton. What is Reasoning? What is an Argument? *Journal of Philosophy*, 87:399–419, 1990.

[186] D. N. Walton. *Argumentation Schemes for Presumptive Reasoning.* Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1996.

[187] D. N. Walton. *The New Dialectic. Conversational Contexts of Argument.* University of Toronto Press, 1998.

[188] D. N. Walton. Justification of argumentation schemes. *Australasian Journal of Logic*, 3:1–13, 2005.

[189] D. N. Walton. Types of dialogue and burdens of proof. In *Proceedings of the 2010 conference on Computational Models of Argument: Proceedings of COMMA 2010*, pages 13–24, Amsterdam, The Netherlands, The Netherlands, 2010. IOS Press.

[190] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning.* SUNY Series in Logic and Language. State University of New York Press, Albany, NY, USA, 1995.

[191] D.N. Walton. *Pactical Reasoning.* Rowman and Littlefield, Savage, Maryland, 1990.

[192] M. Wardeh, F. Coenen, T. J. M. Bench-Capon, and A. Z. Wyner. Multi-agent based classification using argumentation from experience. In J. Zhexue Huang, L. Cao, and J. Srivastava, editors, *PAKDD (2)*, volume 6635 of *Lecture Notes in Computer Science*, pages 357–369. Springer, 2011.

[193] M. De Weerdt, A. Ter Mors, and C. Witteveen. Multi-agent planning: An introduction to planning and coordination. Technical report, In: Handouts of the European Agent Summer School, 2005.

[194] G. Weiss. Achieving coordination through combining joint planning and joint learning. In *ECAI'00*, pages 388–392, 2000.

[195] E. Werner. Toward a theory of communication and cooperation for multiagent planning. In *TARK '88: Proceedings of the 2nd conference on Theoretical aspects of reasoning about knowledge*, pages 129–143, San Francisco, CA, USA, 1988. Morgan Kaufmann Publishers Inc.

[196] J. H. Wigmore. The principles of judicial proof. *The Cambridge Law Journal*, 4:410–411, 10 1932.

[197] M. Wooldridge. Agent-based software engineering. In *IEE Proceedings on Software Engineering*, pages 26–37, 1997.

[198] M. Wooldridge. Semantic issues in the verification of agent communication languages. *Autonomous Agents and Multi-Agent Systems*, 3(1):9–31, 2000.

[199] M. Wooldridge. *Introduction to MultiAgent Systems.* John Wiley & Sons, 2nd edition edition, 2009.

[200] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

[201] M. Wooldridge and N. R. Jennings. The cooperative problem-solving process. *Journal of Logic and Computation*, 9(4):563–592, 1999.

[202] M. J. Wooldrige. *Reasoning about Rational Agents.* MIT Press, 2000.

[203] P. Wyckoff, S. W. McLaughry, T. J. Lehman, and D. A. Ford. T Spaces. *IBM Systems Journal*, 37(3):454–474, 1998.

[204] A. Wyner. Questions, arguments, and natural language semantics. In *Proceedings of the 12th Workshop on Computational Models of Natural Argumentation (CMNA 2012)*, Montpellier, France, 2012.

[205] R. M. Zlot and A. Stentz. Market-based multirobot coordination using task abstraction. In *International Conference on Field and Service Robotics*, July 2003.