An Argumentation-Based Approach to Normative Practical Reasoning

Shams, Zohreh

An Argumentation-Based Approach to Normative Practical Reasoning

submitted by

Zohreh Shams

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Computer Science

December 2015

Abstract

Autonomous agents operating in a dynamic environment must be able to reason about their actions in pursuit of their goals. An additional consideration for such agents is that their actions may be constrained by norms that aim at defining an acceptable behaviour for the agents. The inclusion of normative reasoning into practical reasoning is derived from the necessity for effective mechanisms that regulate an agent's behaviour in an open environment without compromising their autonomy. However, the conflict between agents' individual goals and societal goals (i.e. norms) makes deciding what to do a complicated activity. Argumentation enables reasoning and decision-making in presence of conflict and supports explaining the reasoning mechanism in terms of a dialogue. The need for explanation of a complex task undertaken by an autonomous entity lies in the importance of facilitating human understanding of such entities and consequently increasing their trust in these systems.

Existing argumentation-based approaches to practical reasoning often ignore the role of norms in practical reasoning and commonly neglect the dialogical aspect of argumentation to explain the process of practical reasoning. To address these shortcomings, the research presented in this thesis allows an agent to use argumentation to support deciding what to do while the agent is able to explain why such a decision is made. To this end, we demonstrate a model for normative practical reasoning that permits an agent to plan for conflicting goals and norms. We use argumentation frameworks to reason about these conflicts by weighing up the importance of goal achievement and norm compliance against the cost of goals being ignored and norms being violated in each plan. Such a reasoning serves as the basis of identifying the best plan for the agent to execute, while the reasoning process is explained using natural language translation of a proof dialogue game.

Acknowledgements

I would like to thank my supervisors Dr Marina De Vos and Dr Julian Padget for all their encouragement and patience. Marina's support has never ceased to amaze me, even during her maternity leave. I am so grateful for her excellent supervision on both academic and personal level. I have learnt a lot from Julian during my PhD. Through him I met a lot of academics that have been a great source of inspiration for my research. It has been a great pleasure to work with both of them.

I am indebted to Dr Nir Oren for being my advisor for the past two years. His help has been crucial in the completion and revision of this thesis. I also truly thank Prof. Ken Satoh and Prof. Robert Kowalski for their temporary supervision during my internship in National Institute of Informatics (NII) in Tokyo. I would like to express my appreciation to my examiners Prof. Guy McCusker and Prof. Francesca Toni for their invaluable comments and suggestions.

A big thank goes to my parents, Khatoun and Hamid, who have always been there for me through ups and downs. I also thank my sister and brother, Shirin and Arshia, for bringing joy and happiness to every single moment of my life, including my PhD life. Maman, baba, Shirin and Arshia I love you beyond words, this achievement would not have been possible without your support.

Last but not least are all the lovely people I met through this journey. Pawitra, Fabio, Ali, Swen, Ana, Denise, Tina, Saeed, Fatemeh, Tingting, Gideon, JeeHang and Nataliya you have all been a fantastic part of this journey. I thank you for listening to my worries and standing by my side. Finally, I want to thank my old friends, Roxana, Shooka, Mahdiyeh, Shahrzad, Anna and Naghme for their love and encouragement.

My studies were funded by Graduate School of University of Bath. I would like to use this opportunity to thank them for their financial support. Additionally, the Department of Computer Science at the University of Bath has provided me with financial assistance to attend numerous conferences and research visits for which I am very grateful.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Practical reasoning – reasoning about how to act – for an agent pursuing different goals is a complicated task. Apart from individual goals, agents are often subject to societal norms that encourage the agents to follow the *right* behaviour. A normative practical reasoning agent chooses its actions not only in pursuit of its individual goals, but also according to norms that specify what the agent is obliged to or prohibited from doing under specific conditions. Generating plans for multiple goals while considering norms imposed on the agent is a difficult activity. With the advances made in computational aspects of agent reasoning, autonomous agents are capable of planning under complex conditions. Users often perceive, however, a lack of transparency regarding system outcome due to the intrinsic opacity of agents in open systems. Lack of transparency leads to difficulties in human understanding of the system and its trustworthiness. The main research question in this thesis is *How can we establish transparent mechanisms for autonomous agents to reason about their actions toward satisfying their goals and complying with their norms?*

In this chapter, we present the motivation behind the research presented in this thesis. We also discuss our contributions, as well as the structure of this thesis and the relevant publications.

1.1 Motivation and Problem Statement

Research in the field of autonomous software agents and multi-agent systems has been motivated by the following question since 1980:

> “How do we build agents that are capable of independent, autonomous action in order to successfully carry out the tasks that we delegate to them?” [Wooldridge, 2009, p. 5]

The agents as representatives of individuals and organisations, have proven to be a computationally efficient mechanism to perform many complex tasks in different areas such as electronic commerce, supply chain management and decision support systems. Autonomy as the defining feature of intelligent agents, makes them flexible and enables them to react in different situations through their choices of goals and actions [Norman and Long, 1995]. However, autonomy also poses critical issues about trust, coordination, and reliability unless it is controlled. Controlling and managing the autonomy of intelligent agents has been one of the main challenges of agent systems since their creation. Early research in this area [Moses and Tennenholtz, 1995; Shoham and Tennenholtz, 1992; Walker and Wooldridge, 1995] was mainly focused on introducing some social rules or *conventions* that are hard-wired into the agent and represent a form of internalised control. Being hard-coded into the agents, such conventions are of course the guarantors of reliability and predictability of the agents behaviour. This approach, however, is largely discarded in more recent applications for two reasons. Firstly, introducing sociality in this manner comes at the price of restrictive autonomy. Secondly, the social rules that govern the agents' behaviour are subject to change in response to changes that inevitably happen in a dynamic environment. As a result, the conventions are not necessarily known at design time.

To mitigate the mentioned shortcomings, the research in the area of controlling autonomy shifted from presenting sociality in the form of conventions to sociality in the form of *norms*. The concept of norm in agent societies has its roots in regulative mechanisms in human societies. Normative concepts in human societies and their impact on individuals have been studied for decades in social sciences. From late nineties, these concepts inspired the development of norm-aware entities in artificial intelligence (AI). Norms are social mechanisms that aim at *regulating* agents' behaviour by explicitly specifying obligations, prohibitions, and permissions that apply to the agents under specific circumstances. When implemented as hard constraints [Esteva et al., 2001; Kollingbaum and Norman, 2003; Sadri et al., 2006] an agent has no choice but to comply with the norms. Norms are said to be regimented in this case. However, regimenting norms poses the exact same problems that are raised against hard-wired conventions, namely the huge restriction of autonomy and not being known in advance. Conversely, when implemented as soft constraints, the choice of complying or not complying (i.e. violating) with a norm is left to the agent. In these approaches, known as enforcement approaches, norm compliance is encouraged by introducing consequences in terms of sanctions in case an agent violates the norm [López et al., 2005; Pacheco, 2012; Pitt et al., 2013]. These consequences influence the agent's practical reasoning directly. The agent therefore, must incorporate reasoning about norms and the impact of complying with or violating them, during its practical reasoning.

Since enforcing norms hands the choice of norm compliance over to the agent, it does not restrict the agent's autonomy, but it certainly requires a more sophisticated reasoning from the agent's side. The agent must be able to recognise and reason about any conflict between its individual goals and consequences of norm compliance or violation. Moreover, during its practical reasoning the agent has to reason about the conflict between norms themselves. Conflict between norms is often caused by the agent undertaking different *roles* that require following certain norms that may be conflicting. Alternatively, it can be caused due to the agent interacting with different environments, where these environments enforce their own set of normative objectives. Regardless of the cause of conflict the agent has to consider them in its practical reasoning and act accordingly. However, the question that remains is that when preserving autonomy comes at the cost of such sophisticated reasoning, is the agent still predictable and hence reliable? Is there any solution that conducts practical reasoning, while taking into account the normative position of the agent, such that it is transparent enough to be trusted by humans? This thesis presents an answer to this question. The proposed solution is discussed in the next section.

1.2 Proposed Solution and Contributions

As stated earlier, in contrast to selfish pursuit of individual goals, decision-making about actions in norm-aware agents is also shaped by norms that define right behaviour. Thus, the performance of these agents is not only evaluated by the accomplishment of their goals, but also by respecting the norms of society. As a result these agents' decision-making is more intelligent and human-like [Boella et al., 2006], but it is also more difficult to understand and scrutinise for human users. Explanation plays an important role in making artificial reasoning understandable and thus reliable for human users [Lacave and Díez, 2004; Wooley, 1998]. The explanation capability is in particular useful in convincing the user of an intelligent system about the correctness of the system's results. Lacave and Díez [2004] formally define explaining as:

> "... exposing something in such a way that is *understandable* for the receiver of the explanation – so that he/she improves his/her knowledge about the object of the explanation – and *satisfactory* in that it meets the receivers expectations."

An explanation that meets the user expectation is very likely to have a positive impact on user acceptance. Agents with explanation capability are therefore known to be *persuasive* [Moulin et al., 2002]. In other words, they have a better chance of persuading another agent or a human user to agree with them. Despite the importance, the subject of explanation has received no attention in existing approaches to practical and normative reasoning [Atkinson and Bench-Capon, 2007b; Broersen et al., 2001; Criado et al., 2010; Hulstijn and van der Torre, 2004; Kollingbaum and Norman, 2003; Rahwan and Amgoud, 2006; Sadri et al., 2006].

Argumentation serves as an effective computational tool for various agent activities including agent reasoning [Amgoud, 2003; Bench-Capon et al., 2009; Dung, 1995; Gaertner and Toni, 2007b; Oren et al., 2007]. As a reasoning tool, argumentation is particularly important because it allows drawing consistent conclusions from a set of conflicting, inconsistent and incomplete information [Bench-Capon et al., 2009; García et al., 2013]. The process of argumentation [Amgoud et al., 2004] consists of (i) building a set of arguments; (ii) identifying their conflicts, referred to as *attacks*; and (iii) recognising the *acceptability* of arguments based on their weights, the attacks they receive, and the counter-attack they present. Based on the acceptability of arguments, certain conclusions can be drawn and used for different purposes including agent's internal reasoning or multi-agent collaborative reasoning. Dung [1995] proposed one of the most widely used *argumentation framework* that is the basis for most of research in argumentation-based reasoning. An argumentation framework consists of a set of arguments and a set of attacks between them: $AF = \langle Arg, Att \rangle, Att \subseteq Arg \times Arg$. Various acceptability criteria, known as argumentation *semantics*, were also proposed by Dung [1995] to identify the status of arguments in an argumentation framework.

In addition to reasoning based on argumentation frameworks, argumentation can also serve as an effective computational tool for generating explanation [Baroni and Giacomin, 2009; Caminada et al., 2014c; Fan and Toni, 2015; García et al., 2013; Lacave and Díez, 2004]. Intelligent agents equipped with argumentation capabilities can explain the validity of their recommendation to their users in a form of explanatory dialogues that are similar to human argumentation activities [Moulin et al., 2002]. These dialogues formalise dialectical explanation support for argumentation-based reasoning based on argumentation semantics. However, in contrast to semantics that justify the validity of an argument in terms of membership of a set, these dialogues, through some fictitious *proponent* and *opponent* dialogue game [Fan and Toni, 2015] provide dialectical explanation for valid arguments.

Going back to our main research question posed in the previous section "Is there any solution that conduct practical reasoning taking into account the normative position of the agent, such that it is transparent enough to be trusted by humans?" we seek the solution in argumentation-based approaches to reasoning. Because not only does argumentation deal with conflicts and inconsistencies as a part of reasoning toward decision-making, it is also explainable in the form of a dialogue. Consequently, normative practical reasoning can be conducted in a transparent way that is likely to be trusted by human users.

In this thesis we adopt the viewpoint of Pollock [1995] to practical reasoning, where firstly, plans are generated with respect to what the agent cares about and secondly, the plans are subject to decision-making based on certain criteria and considerations. Plans for the agent are defined and generated with respect to the agent's individual goals and the norms imposed on the agent. Each of the generated plans are seen as proposals of actions that need to be evaluated in order for the agent to identify the best plan. We use *argumentation schemes* and *critical questions* [Walton, 1996] to equip the agent with the ability to question, defend and reject the plan proposals. Schemes are general patterns of arguments expressed in natural language and there is a set of critical questions associated with each scheme that presents the ways in which the scheme can be attacked.

The argumentation framework built for each plan proposal is evaluated to identify the justified plans. The justified plans are further compared based on the set of goals they satisfy/do not satisfy and the set of norms they comply with/violate. The explanation of the best plan is demonstrated through the representation of justifiability in terms of argumentation-based dialogues. In order to provide an explanation that is natural, clear and easy to understand, the explanation is translated into natural language.

In summary, the main aim of this thesis is in setting out an end-to-end solution that takes agents actions as input and equips the agent with the ability to act in the presence of conflicting goals and norms, while allowing the agent to explain why it acted as it did. In achieving this aim we propose the first argumentation-based approach to practical reasoning that uses both the reasoning and explanation capability of argumentation. In so doing, the following contributions are made:

- Formalising a novel model for normative practical reasoning that defines plans considering multiple goals and norms, while taking into account their conflicts.

- Implementing the model to automate generating the plans.

- Creating and formalising a set of argument schemes and critical questions that integrate norms and durative actions into practical reasoning; These schemes are aimed at checking the justifiability of plans with respect to goals satisfied and norms complied with or violated in the plan.

- Offering a novel decision criterion that identifies the best plan, taking into consideration the justifiability of plans and preferences over goals satisfied and norms violated across plans. In absence of sufficient preference information, the number of goals satisfied and norms violated is the basis of plan comparison.

- Proposing a concrete application of a recently developed dialogue game [Caminada et al., 2014b] that dialectically explains why a plan is justified.

- Providing a natural language explanation for why a plan is the best plan, such that the explanation is easy to understand for experts and non-expert users.

1.3 Thesis Outline

This thesis is structured as follows:

Chapter 1: This chapter addresses the motivation behind this research, as well as the contributions that this work makes. It also provides an overview of the work presented in the remaining chapters.

Chapter 2: Related literature to this work are surveyed in this chapter. We first give an overview of agent reasoning in general, followed by practical reasoning and the role of norms in practical reasoning. Subsequently, different approaches to normative practical reasoning are discussed. In particular, argumentation-based approaches to practical reasoning are surveyed in detail. It is explained how argumentation can support agent reasoning and human understanding of such reasoning.

Chapter 3: Proposing a formal model for normative practical reasoning is the focus of this chapter. This model permits the agent to plan for multiple goals and norms, while considering their conflicts. Conflict between actions, goals and norms are explicitly presented and formulated. Plans are consequently defined with respect to these conflicts.

Chapter 4: An implementation of the formal model in the previous chapter is presented in this chapter. The implementation is aimed at automating the generation of plans defined in the formal model. The computational tool for the implementation is answer set programming [Baral, 2003]. The absence of a conceptual gap between the formal and computational model, is one of the primary advantages of the implementation.

Chapter 5: Chapters 3 and 4 constitutes the first step of practical reasoning, namely planning. The second step of practical reasoning, namely decision-making about which plan to execute is what we address in this chapter. Argumentation frameworks are used to assist the agent's decision-making about plans. A formal model of arguments based on argumentation schemes and the relationships between arguments based on critical questions are presented in this chapter. Preferences are used to reflect the weight of arguments. They are taken into account in determining the acceptability of arguments and ultimately identifying the best plan.

Chapter 6: The explanation of why a certain plan should be executed is provided in this chapter. Argumentation-based dialogues are used to give a human-like explanation of the reasoning process toward identifying the best plan. The dialogue is translated to natural language, which makes it readily comprehensible.

Chapter 7: Conclusions and contributions of this work are addressed in this chapter. The limitations of this research are also pointed out, with possible solutions to tackle them. Also, further research directions are discussed.

1.4 Related Publications

Some part of this thesis is published in the following papers:

- Shams, Z., De Vos, M., Oren, N., Padget, J., *Explaining Normative Practical Reasoning via Argumentation and Dialogue*, Submitted to International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016).
  This paper contributed toward Chapters 5 and 6.

- Shams, Z., De Vos, M., Oren, N., Padget, J., and Satoh, K., *Argumentation-based Normative Practical Reasoning*, accepted for publication in Proceedings of International Workshop on Theory and Applications of Formal Argument (TAFA 2015).
  This paper contributed toward Chapters 5 and 6.

  In this paper we proposed a model for normative practical reasoning that allows an agent to plan for multiple and potentially conflicting goals and norms at the same time. The best plan for the agent to execute is identified by means of argumentation schemes and critical questions. The justification of this choice is provided via an argumentation-based persuasion dialogue for the grounded semantics.

- Shams, Z., De Vos, M., Padget, J., and Vasconcelos, W., *Implementation of Normative Practical Reasoning with Durative Actions*, accepted for publication in Proceedings of International Workshop on Coordination, Organisation, Institutions and Norms in Multi-Agent Systems (COIN 2015).
  This paper contributed toward Chapters 3 and 4.

This paper proposed a formal model that allows the agents to plan for conflicting goals and norms in presence of durative *actions that can be executed* concurrently. *Plans are compared based on decision-theoretic notions (i.e. utility) such that the utility gain of goals and utility loss of norm violations are the basis of this comparison. The set of* optimal *plans consists of plans that maximise the overall utility, each of which can be chosen by the agent to execute. The formal model is implemented using answer set programming, which in turns permits the statement of the problem in terms of a logic program that can be queried for solutions with specific properties. It is demonstrate how a normative practical reasoning problem can be mapped into an answer set program such that the optimal plans of the former can be obtained as the answer sets of the latter.*

- [Shams, 2015] Shams, Z. (2015), *Normative Practical Reasoning: An Argumentation-Based Approach*, In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015), pages 4397–4398.

  The extended abstract presented in this paper provides a summary of the thesis.

- [Shams et al., 2013] Shams, Z., De Vos, M., and Satoh, K., *ArgPROLEG: A Normative Framework for the JUF Theory*, New Frontiers in Artificial Intelligence, Springer, 2013, pages 183–198.
  This paper contributed toward Chapters 2 and 5.

  In this paper we proposed ArgPROLEG, an application of argumentation theory in legal reasoning. This application is based on PROLEG, an implementation of the the Japanese "theory of presupposed ultimate facts" (JUF). This theory was mainly developed with the purpose of modelling the process of decision-making by judges in the court. Not having complete and accurate information about each case, makes uncertainty an unavoidable part of decision-making for judges. In the JUF theory each party that puts forward a claim, due to associated burden of proof to each claim, it needs to prove it as well. Not being able to provide such a proof for a claim, enables the judges to discard that claim although they might not be certain about the truth. The framework that we offered benefits from the use of argumentation theory to allow reasoning with incomplete and inconsistent information. Furthermore, it brings the reasoning closer to the user by modelling legal rules in terms of normative concepts.

Chapter 2

Literature Review

The aim of this work is to enable an agent operating in a normative environment to identify and justify the best course of actions to execute. Reasoning involved in deciding what to do is referred to as *practical reasoning*. While there are different approaches addressing the practical reasoning problem in agents, we claim that they often lack transparency when it comes to explaining and justifying this reasoning process and its results. Argumentation has proven to be a promising approach in aiding agent's reasoning and decision-making in a scrutable and trackable way [Caminada et al., 2014c; Fan and Toni, 2015; Kakas and Moraitis, 2003; Oren, 2013; Zhong et al., 2014]. To this end, we propose an argumentation-based framework for practical reasoning based on argument schemes and critical questions [Walton, 1996]. To put this work into context, in this chapter, we provide a summary of related work addressing argumentation-based practical reasoning and planning. Section 2.1 gives an account of (i) practical reasoning in general, (ii) the role of norms in an agent society, and (iii) practical reasoning in the presence of norms. In Section 2.2 we focus on the role of argumentation in agent reasoning and agent dialogue for making and explaining decisions. Section 2.3 discusses the contributions and limitations of existing approaches that use argumentation as the basis of practical reasoning. Finally we conclude in Section 2.4 by describing the intuition behind this research and how it relates to the existing approaches.

2.1 Agent Reasoning

Theoretical or epistemic and *practical* reasoning are two known elements in rational agents' reasoning and decision-making. Epistemic or theoretical reasoning is concerned with what to believe, whereas practical reasoning is reasoning about actions and what to do according to some motivational attitudes, such as goals, desires or intentions [Bratman, 1987; Wooldridge, 2000]. The similarities and differences between reasoning about beliefs and actions have been much discussed in the past [Fox and Parsons, 1998; Pollock, 1995; Wooldridge and Jennings, 1995]. In this section we review works by the three most influential scholars amongst others, who studied the distinctions between agent epistemic and practical reasoning, namely Pollock, 1995, Walton, 1996 and Searle, 2001.

Pollock [1995], as one of the first scholars who studied agent reasoning, appreciates the distinction between epistemic and practical reasoning, but also expresses the difficulty of making a precise distinction between the two due to their interdependency. In his view, practical reasoning must be based on the agent's beliefs of the current situation and beliefs are of course the result of epistemic reasoning. On the other hand, the whole point of epistemic reasoning is to address the agent's practical problems. Reasoning about beliefs does not happen at random, it has to be triggered by questions that are posed by practical reasoning. As a result of this viewpoint, he built the well-known agent architecture, *OSCAR* [Pollock, 1995], that handles epistemic and practical reasoning simultaneously. Walton [1996] sees the main distinction between theoretical and practical reasoning in their distinguished aims. The aim in a theoretical inference is to establish the truth or falsity of a proposition, whereas, in a practical inference the aim is to get from the agent's premises in terms of its current situation and goals, to the imperative conclusions that direct the agent towards a prudent course of action. Searle [2001] also believes that reasoning about actions differs from epistemic reasoning and needs additional features compared to the other. He enumerates these three features as follows:

(i) *first-personal:* reasoning about belief is universal. A proposition being proven true, is a reason for anybody to believe that is the case. But reasoning about actions is subjective and depends on the agent's motivational attitude such as goals and desires.

(ii) *future-directed:* reasoning about action is tied up to time, which is not the case when reasoning about beliefs. More precisely, reasons for acting are forward-looking.

(iii) *motivational:* reasons for action need to have a motivational essence that essentially motivates taking an action.

In addition, Searle takes the discussion on differences between theoretical and practical reasoning further by emphasising that

> ... "theoretical reason is a special case of practical reason: deciding what beliefs to accept and reject is a special case of deciding what to do." [Searle, 2001, p. 136]

However, Pollock [1995] argues that viewing epistemic reasoning as a special kind of practical reasoning leads to infinite regress, since he believes that any practical reasoning must be based on some epistemic reasoning, even if not modelled explicitly.

To summarise, although the viewpoints about distinctions between practical reasoning and theoretical reasoning are various, they unanimously agree that these two types of reasoning have to be treated differently. Despite this, many argue that practical reasoning has not been studied within computer science or philosophy nearly as extensively as reasoning about beliefs has. Much research in early days of AI mainly focused on theoretical reasoning, however the growth of software agent technologies demanded agents that can conduct practical reasoning or in other words agents that are capable of reasoning about actions and deciding what to do [Atkinson, 2005]. In this thesis we focus on the practical aspects of agent reasoning. The next section gives an overall view of agent practical reasoning.

2.1.1 Practical Reasoning

In the previous section we explained that practical reasoning is reasoning towards actions; a type of reasoning that results in deciding what to do. But, solving decision problems using decision theory [Jeffrey, 1983; Savage, 1954] has been conceptualised and formalised in the past. Why use practical reasoning to make a decision? This section aims to answer this question and define practical reasoning in agent systems.

When conducted for the purpose of enabling an agent to decide what to do, the similarity between decision-making and practical reasoning is clearly in aiming to solve the same problem: *what to do?!* However answering the question of what to do, requires the agent to know what courses of actions (i.e. plans) are available in the first place. Classical decision theory [Jeffrey, 1983; Savage, 1954], that was originally developed within economics and forms the foundation of decision-making, does not have much to say in this respect. What decision theory does is comparing a set of alternatives by means of a decision criterion. The alternatives are assumed to be given and where they come from is not the focus of decision theory at all. Pollock [1995] is a strong advocate of the inadequacy of decision-theoretic models in complex decision problems that require planning. One of the examples that Pollock uses to demonstrate this inadequacy is illustrated in Figure 2-1. Assume that pushing the top four buttons in Figure 2-1 gives a utility of 10, while pushing the bottom button produces a utility of 5. Evidently, pushing the top four buttons gives a better utility, so based on decision theory that is what one ought to do. But, the decision-theoretic model applies to actions and it only allows us to compare the expected utility of *pushing button A* and *pushing button B*. However, the expected utility of *pushing button A* is not apparent to us, since the single act of pushing this button does

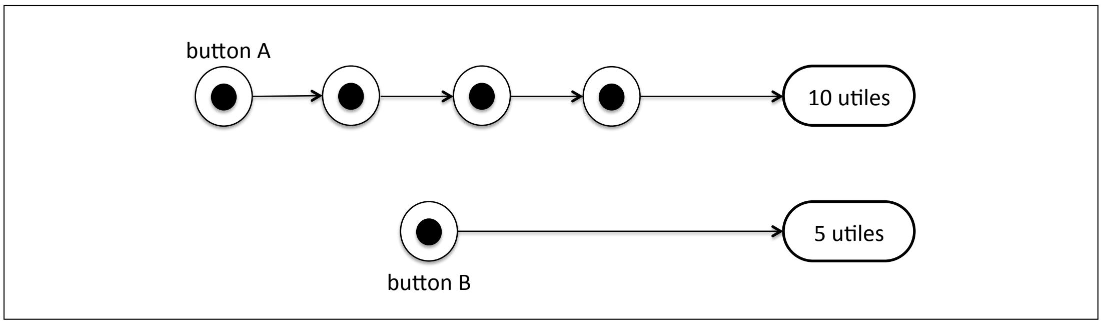


Figure 2-1: Counterexample of Decision-theoretic Model [Pollock, 1995, p. 179]

not generate the utility of 10, unless it is followed by pushing the other three buttons. Calculating the utility of *pushing button A* requires us to consider the probability of pushing the rest of the buttons in that row. Depending on these probabilities, the expected utility of *pushing button A* might well be less that *pushing button B*. Thus, applying the decision-theoretic notions to the whole plan, paradoxically, prescribes *pushing button B* rather than *pushing button A*.

The point that Pollock tries to make through a set of examples such as the one we just discussed, is that when decision-making requires planning, decision theory does not necessarily provide us with a rational answer. *He therefore argues that what is needed in such problems is a practical reasoning theory that defines plans and applies decision-theoretic concepts to plans rather than actions.* Furthermore practical reasoning has the added value of being explainable. Although decision theory provides the best alternative out of a set of alternative, it does not justify why that is the case. However, conducting practical reasoning in a way that is explainable and justifiable is the main concern in several recent literature (e.g. [Amgoud et al., 2008b; Oren, 2013]).

Having explained the distinction between using a combination of practical reasoning and decision theory as opposed to a solely decision theoretic solution, we now give a more elaborate account of practical reasoning. Practical reasoning has been defined in several various ways and from different perspectives. In this thesis we only discuss those definitions that have been widely used in the AI community. One of the most popular definitions of practical reasoning that is often referred to in AI is given 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." [Bratman, 1990, p. 37]

Later on Bratman's definition of practical reasoning was extended by Searle [2001] to a process that not only considers and weighs all possible options but also tries to identify the best of them. Studying this definition gives a clear picture of how decision-making contributes to the process of practical reasoning, where the former defines a set of alternatives and the latter evaluates the alternatives based on some decision criterion. A more recent definition of practical reasoning by Wooldridge [2000], that is adopted in this thesis, recognises two distinct activities for a practical reasoning agent, namely *deliberation* and *means-ends reasoning*. Deliberation is the process of identifying which goals to pursue, followed by *means-ends reasoning*, in which the agent searches for plans to satisfy those goals. If there is a plan for satisfying a goal, that goal is considered *feasible*. Consequently, in the absence of such a plan, not all of agent's goals are feasible. Moreover, the existence of plans for every single goal does not guarantee satisfying all of them, due to for example, conflicting use of resources between the plans. Although the idea of modelling practical reasoning in two steps was well received and implemented in a number of works (e.g. [Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006]), conducting the two steps separately did not prove to be as successful. The criticism against this separation is given in Amgoud et al. [2008a], where the authors argue that by this separation the agent might commit to some goals at the deliberation step that are not feasible, due to (i) lack of a plan to achieve them in the means-ends reasoning step and/or (ii) existence of conflicting plans. Thus, Amgoud et al. [2008a] debate that when satisfying all of the agent's goals is not possible, a practical reasoning agent should seek the subsets of goals along with their plans that are both feasible and consistent.

A summary of this section is demonstrated in Figure 2-2 that illustrates the link between practical reasoning theory and decision theory. This figure shows how a set of alternatives resulting from practical reasoning are evaluated based on decision-theoretic notions. The decision criterion can be expressed in several ways. In classical decision theory (CDT) the decision criterion is expressed through notion of utility, whereas, in qualitative decision theory (QDT), it is expressed in terms of preference information. Dastani et al. [2005] make the following distinctions between CDT and QDT:

- The underlying concepts in CDT are probability function, utility function and decision rule, whereas likelihood ordering, preference ordering, and decision criterion are the key concepts in QDT.

- In CDT a good decision is defined as a decision that maximises the expected utility, while in QDT a good decision is characterised as a decision that most satisfies the decision criterion.

- QDT is often computationally more efficient than CDT, however QDT provides a pref-

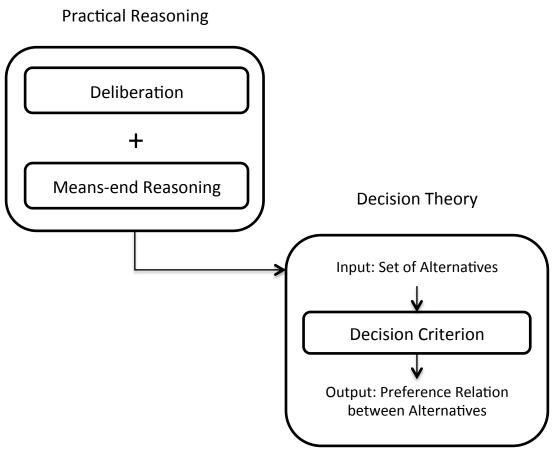


Figure 2-2: Practical Reasoning and Decision Theory

erence relation between choices without the measurement of how much one decision is preferred over the other one. In CDT the difference in utility of options is easily measurable.

Fox and Parsons [1998] argue that in the context of practical reasoning, the benefits of CDT over QDT can be small by comparison with the restrictions imposed by the formalism. Atkinson [2005] summarises the main concern raised in Fox and Parsons [1998] as the impracticality of generating a set of probabilities and utilities that is demanded by CDT. Requiring less quantitative information, Fox and Parsons [1998] recommend QDT as an alternative. Prakken [2006a] reinforces this argument by stating that often a decision maker, for instance an agent, has only partial and qualitative information about probability and preference rather than the precise information required in CDT. In addition to the above arguments, Doyle and Thomason [1999] mention the following reasons for why CDT is not adequate in realistic cases:

1. CDT does not address making decisions in unforeseen circumstances or when decision-making involves a broad knowledge of the world.

2. CDT cannot capture generic preferences that are common human expressions in a convenient formal manner.

3. CDT offers no means to help decision makers who exhibit discomfort with numeric trade-offs.

The analysis of comparison of CDT and QDT has led us to choose to use qualitative preference ordering to express an agent's priorities over goals and norms. We will return to this point in Chapter 5.

2.1.2 Normative Practical Reasoning

Managing the autonomy of intelligent agents has been one of the main challenges of agent systems since their creation. Many [Jennings, 1993; Moses and Tennenholtz, 1995; Shoham and Tennenholtz, 1992; Walker and Wooldridge, 1995] have sought the solution in approaches that influence the agent behaviour externally. Introducing constraints or *conventions* at design time in social systems was one of the early solutions to influence agent reasoning [Moses and Tennenholtz, 1995; Shoham and Tennenholtz, 1992; Walker and Wooldridge, 1995]. Walker and Wooldridge [1995] define conventions as:

> "... a behavioural constraint, striking a balance between individual freedom on the one hand, and the goal of the agent society on the other hand." [Walker and Wooldridge, 1995, p. 1]

However such conventions were hard-wired into the agents which hugely restricted the autonomy and flexibility that are the central features of autonomous agents. Consequently, more flexible approaches [Conte et al., 1999; Shoham and Tennenholtz, 1997] were investigated in which conventions have *regulative* roles rather than restrictive. *Norms* are social mechanisms that, depending on the way they are implemented, can regulate agent behaviour without compromising its autonomy and therefore play a very important role in an agent's practical reasoning. In order to discuss normative practical reasoning, we first need to answer the following questions.

- What are norms?
- How norms are specified and implemented?
- How do agents reason about norms?

What Are Norms?

Managing the autonomy of agents without compromising their autonomy is a challenging task. Norms have been introduced and implemented as a solution that can influence the reasoning and decision-making of agents toward actions in different ways. The definition offered by the Merriam-Webster dictionary explicitly mentions the regulative aspect of norms in any group or society:

> "Norms are principles of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behaviour."

Answering the question of how norms express such control or regulation has yielded different classifications of norm [Boella and van der Torre, 2008]. One of the common classifications distinguishes between constitutive and regulative/behavioural norms [Boella and van der Torre, 2004]. The former category aims at creation of institutional fact that describe the legal consequences of actions in a normative system, whereas the latter category aim at defining an ideal behaviour and regulating agents by expressing what is *obligatory, forbidden or permitted* [López and Luck, 2003]. In this thesis we focus on behavioural norms that are externally imposed on the agent with the aim of regulating its autonomy.

Depending on the type of society modelled, the agent's behaviour is regulated by different type of norms. In *permissive* societies the agent is allowed to perform any action or achieve any state unless it is explicitly forbidden. While in *prohibitive* societies the agent is not permitted to do anything unless specified. *Obligations, prohibitions and permissions* are the common norms in permissive societies, whereas *Power, permission and obligation* are common norms in prohibitive societies [1]. Obligation norms in both cases dictate what the agent is obliged to do; power norms denote the capability of doing something; prohibition norms express what is forbidden in permissible societies; conversely, permission norms express what is allowed in prohibitive societies. However, permission norms have been treated differently in permissive societies. Some work, [Alrawagfeh and Meneguzzi, 2014; Kollingbaum, 2005], consider them as an explicit statement that allows the agent to execute an action. In Alrawagfeh and Meneguzzi [2014], permissions are utilised when the agent does not have complete knowledge about the environment it operates in. In such an environment executing an action that is explicitly permitted is always safer than an action that is not stated as permitted, because it may have been forbidden. Some other approaches utilise permission norms to model the exceptions for obligation and prohibition norms [Oren, 2013; Oren et al., 2010; Pacheco, 2012]. In such cases, the agent is obliged to or prohibited from executing an action unless there is a permission norm that permits not executing the obliged action or it permits the execution of a forbidden action.

How Norms Are Specified and Implemented?

Following the above background on norms, we now turn our attention to their specification and implementation. In order to be able to influence the agent's behaviour, norms have to be explicitly specified and presented to the agent. The agent can then use its normative reasoning capability to decide how to adapt its behaviour according to the imposed norms. Many nor-

[1]Obligations, prohibitions, power and permissions can be applied on actions or states. Since action-based norms are the focus of this thesis, in the rest of this document we only refer to these operators being applied to actions.

mative languages have been proposed [Dastani et al., 2009; García-Camino et al., 2005; Oren et al., 2008; Uszok et al., 2008; Vázquez-Salceda et al., 2004] to present and specify norms, a comprehensive survey of which can be found in [Pacheco, 2012]. Debating the similarities and differences of properties of these languages is not within the scope of this thesis, therefore, we just briefly mention five elements that differentiate languages for specifying norms, identified by Pacheco [2012]:

1. *Deontic Operators* that define the type of normative proposition, O (obligation), F (prohibition) and P (permission), modelled.
2. *Controls* that determine whether the deontic propositions operate on actions, states or both.
3. *Enforcement Mechanisms* that show whether sanctions and/or rewards are used to enforce norm compliance.
4. *Conditional Expression* that indicates whether the norm activation condition is an action and/or a state.
5. *Temporal Constraints* that specify constraints on norm activation or termination such as "before", "after" or "between".

We will return to these elements in the next chapter (page 60) and describe how the normative language we use fits with them.

Regardless of the presentation and specification, the mere explicit representation of norms is not sufficient to expect an agent to recognise them. Norms should be implemented in a way that the agent can recognise when they are activated and violated and what the consequences of a violation are. Figure 2-3 sets out a taxonomy of norm implementation mechanisms. They are divided into two categories: *regimentation* and *enforcement*. In regimentation approaches [Esteva et al., 2001] norms are modelled as hard constraints and the agent has no choice but to blindly follow the norms. Conversely, in enforcement approaches norms are modelled as soft constraints leaving the choice of complying or not complying to the agent. But, in order to encourage norm compliance, there are consequences introduced in terms of sanctions in case the agent violates the norm [López et al., 2005; Pitt et al., 2013]. Moreover, in some enforcement approaches [Aldewereld et al., 2006] the agent is rewarded for complying with a norm. The regimentation approaches are further divided into *mediation*, in which there exists a reliable entity that prevents the agent from violating the norms ; and *hardwiring*, in which the agent's mental attitude are manipulated in accordance with norms. On the other hand, enforcement approaches are classified based on the entity in charge of norm enforcement. If

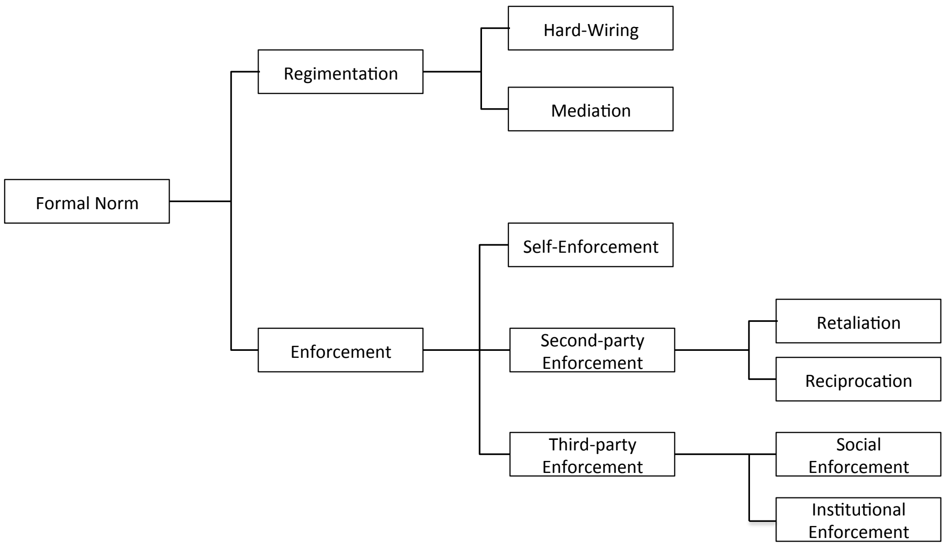


Figure 2-3: Norm Implementation Mechanism [Pacheco, 2012, p. 37]

the agent itself is in charge of sanctions in case it violates a norm, the approach is said to be *self-enforcement*. In *second-party norm enforcement*, each party in the transaction is in charge of norm enforcement for other parties by applying reward or punishment (*retaliation*), or by returning similar interchange to the one presented (*reciprocation*). In the last approach, *third-party norm enforcement*, a judge or an authority external to the agents enforces the norm. In *social enforcement* this authority is the society, whereas in *institutional enforcement* infrastructural entities called institutions act as norm enforcer by defining institutional sanctions against norm violators.

How Do Agents Reason about Norms?

To answer the question of how an agent reasons about norms, we need to look back to what we discussed earlier about norm implementation methods. If the norms are regimented, the agent does not need to reason whether it wants to obey them, because it is forced to do so. On the other hand, in enforcement approaches, which are the focus of this thesis, the agent has the choice to comply with the norms or not. The question is how the agent decides in favour of or against obeying a norm? Decision-making on norm compliance has received a lot of attention in the past ten years or so. Here we provide a survey of those approaches that consider the question of whether to comply with a norm, in the context of practical reasoning and planning.

The BOID (Belief-Obligation-Intention-Desire) architecture [Broersen et al., 2001] extends the BDI architecture [Rao and Georgeff, 1995] with the concept of obligation and uses agent types such as social, selfish, etc. to handle the conflicts between beliefs, desires, intentions and obligations. For instance if the agent is selfish, it will always considers its desires prior to any obligation. In contrast, a social agent always puts obligations prior to its desires. This architecture is considered as a model for norm-governed agent, although it lacks a computational model for implementation.

NoA, proposed by Kollingbaum [2005], is a normative language and agent architecture. As a language, it specifies the normative concepts of obligation, prohibition and permission to regulate a specific type of agents interaction called "supervised interaction". As a practical reasoning agent architecture, it describes how agents select a plan from a pre-generated plan library such that the norms imposed on the agent at each point of time are obeyed. The agents do not have internal motivations such as goals or values that might conflict with norms, which therefore, enables the agent always to comply with norms. However, there may be a conflict between norms imposed on the agent and hence the need for conflict resolution mechanisms such as those proposed in this work.

López et al. [2005] propose a normative framework for goal-driven agents in which the agents are persuaded to obey norms if not complying with a norm hinders an agent's individual goals. Compliance therefore, relies on the explicit interaction between goals and norms. If norm compliance or violation does not hinder any goal, there is no connection and hence no computational mechanism in place that enforces norms. Figure 2-4 shows how the agent deals with the dilemma of complying with a norm or not. When there is a conflict between a norm and the agent's goals, the agent does not comply unless the goals hindered by punishment are more important than goals facilitated by compliance. On the other hand, if there is no such conflict, the agent only complies with a norm if there are goals that are hindered by the punishment of violation, and violates it otherwise.

Sadri et al. [2006] extend the KGP (Knowledge, Goals and Plans) model of agency [Kakas et al., 2004] to support agent normative reasoning based on agent roles. They claim that defining roles for the agents along with obligations and prohibitions that result from playing various roles, enables KGP agents to generate plans for their goals while reacting to changes in the dynamic environment in which they are situated. Furthermore, they argue that although their proposed approach considers norms within an individual social agent, it is scalable to multi-agent systems that are organised through norm utilisation. One of the advantages of this model is the conflict detection mechanism between agents' individual goals and norms, however this model lacks a conflict resolution mechanism. Essentially, in case of conflict, a KGP agent follows the norm imposed on it rather than its internal goals.

Oren et al. [2011] take norms into consideration when deciding how to execute a pre-generated plan with respect to the norms triggered by that plan. Plans in the agent's plan library

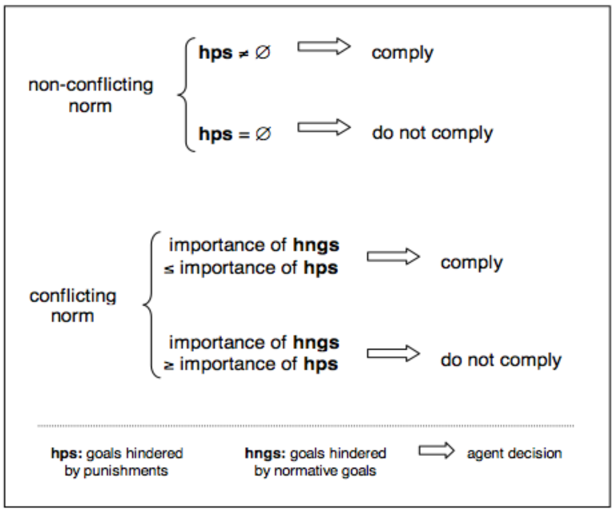

non-conflicting norm
hps ≠ ∅
comply
hps = ∅
do not comply
conflicting norm
importance of hngs ≤ importance of hps
comply
importance of hngs ≥ importance of hps
do not comply
hps: goals hindered by punishments
hngs: goals hindered by normative goals
agent decision

Figure 2-4: Pressured Norm Compliance [López et al., 2005, p. 10]

are designed to satisfy the agent's individual goals and cannot possibly take into account all the environmental variables, such as norms, that may influence the agent's behaviour at run time. Thus, pre generated plans need to be adjusted to cater for norms imposed on the actions in the plan at each point in time. A norm imposed on an action intends to constrain the values assigned to some variables within that action. The adjustments of values in actions with respect to norms imposed to the actions, aim to specify how the agent should execute a plan such that the cost of violated norms is outweighed by the reward obtained from norms complied with. The most preferred plan is the one that maximises the utility.

In Panagiotidi et al. [2012b], the authors argue that most of the frameworks that accommodate norms in practical reasoning are focused on goal or plan selection and there has not been enough attention paid to incorporating norms into the agent's plan generation. They therefore, propose a norm-oriented agent in which norms are taken into account in the agent's plan generation phase. To this end, they introduce a "norm-aware planner" that checks the normative state of the agent after each individual action is taken. The planner then decides if the agent should comply with a norm or not based on the agent's utility function over the actions. Although this mechanism enables the agents to cope with the dynamics of operating in an open environment, checking the state of agent after each action, depending on the number of actions, imposes a high computational cost on the plan generation phase.

The approaches reviewed above, explore different strategies to handle normative practical reasoning when conflict arises between mental attitude of the agent including beliefs, goals, desires, norms and plans. Some only focus on conflict between goals/desires or conflict between norms, while others concentrate on conflict between goals and norms simultaneously. Generally speaking, in these approaches, there has not been much attention paid to explaining the agent decision-making process. Argumentation not only allows reasoning in the presence of conflict, but it also permits the presentation of the reasoning process in terms of a dialogue that – even for non-expert users – is easy to follow. For this reason, there has been an increasing trend of using argumentation in practical reasoning and practical reasoning in presence of norms. In the next section we investigate how argumentation can contribute to the agent's reasoning and decision-making and the explanation of these processes. Since argumentation-based normative practical reasoning is the focus of this thesis, these approaches are discussed in details in section 2.3.

2.2 Argumentation

The theory of argumentation dates back to Greek philosophy and since then it has been cultivated in many research areas such as psychology, law, communication studies, and artificial intelligence. The study of argumentation as a process primarily emerged in interpersonal communication studies in the seventies, where argumentation was and is mainly studied as a verbal and social activity. Argumentation as a social activity enables people to argue for all sorts of reasons including to justify their thinking, to defend their actions or perspectives, to judge and decide in controversial situations, and so on. Apart from being a social activity, argumentation is an intellectual and rational activity aimed at establishing the legitimacy of a standpoint by bringing arguments justifying or refuting the original standpoint [Eemeren et al., 1996]. Arguments themselves are claims supported by reasons, and reasons are supported by evidence themselves. The defeasible nature of inference from evidence to reasons and from reasons to claims has made argumentation widely accessible in non-monotonic reasoning [Dung, 1995; Pollock, 1992; Simari and Loui, 1992; Vreeswijk, 1992]. On the other hand, the dialectical nature of the argumentation process [Bentahar et al., 2004; Caminada and Podlaszewski, 2012b; Hamblin, 1970, 1971] has made it a common choice to model dialogues taking place for different purposes such as making agreement, negotiation, and etc. [Amgoud and Vesic, 2012; Kraus et al., 1998]. The accessibility of argumentation in modelling defeasible reasoning and dialogues have both been exploited in the field of agent reasoning and multi-agent systems. In Sections 2.2.1, 2.2.2, and 2.2.3 we explore the role of argumentation for agent reasoning, agent dialogues, and generating explanation respectively.

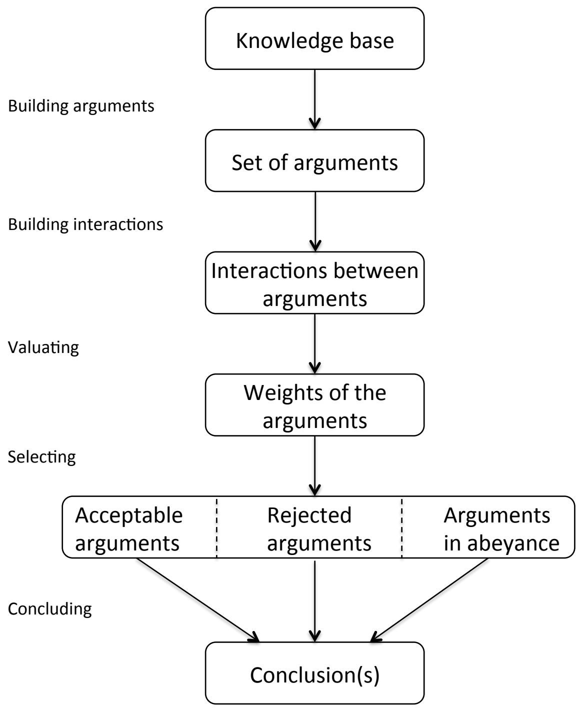


Figure 2-5: The Process of Argumentation [Amgoud et al., 2008c]

2.2.1 Argumentation for Agent (Non-monotonic/defeasible) Reasoning

Argumentation theory, for the purpose of non-monotonic reasoning, aims at forming a set of arguments that are collectively acceptable, such that from this set coherent and justified conclusions can be drawn. Figure 2-5 highlights the process of argumentation in five steps:

1. Building a set of arguments from a knowledge base;
2. Identifying the interactions between arguments;
3. Valuating (sic) the weights of arguments;
4. Recognising the position of arguments based on their interactions and weights in terms of accepted arguments, rejected ones and those that are undecided; and
5. Concluding what are the set(s) of justified arguments.

Using argumentation as the basis of non-monotonic reasoning goes back to Lin and Shoham [1989] and Pollock [1992]. His work was then cultivated and culminated by Simari and Loui [1992]; Vreeswijk [1992] and Dung [1995], respectively. The work of Dung [1995] on *argumentation frameworks (AF)* is the foundation of most of today's work on argumentation theory. Dung's argumentation framework (DAF) is formally defined as a pair: $AF = \langle Arg, Att \rangle$, where Arg is a set of arguments and $Att \subseteq Ar \times Ar$ is the attack relation between arguments. Argument A attacks argument B iff $(A, B) \in Att$. But how does establishing arguments and their relations in an AF result in identifying a set or sets of coherent arguments? Answering this question, gave rise to the concept of *argumentation semantics*, which are criteria to determine a set of justified and coherent arguments based on argument interactions. If two arguments attack each other then an entity – which could be an agent for example – cannot accept both of them at the same time. Therefore, argumentation semantics are there to examine the acceptability of a set of arguments. We now give the definition of the four main semantics that Dung introduced [Dung, 1995], namely the *complete, grounded, preferred* and *stable* semantics. But first, we define the concepts of *conflict-freeness, acceptability* and *admissibility* for a set of arguments.

Conflict-freeness: A conflict-free set is a set in which none of the arguments attacks another. This is the minimum criteria for a set of arguments to be considered as coherent.

Acceptability: An argument P is said to be acceptable with respect to set $\mathcal{S}$, iff $\forall\ A, (A, P) \in Att \Rightarrow \exists\ Q \in \mathcal{S}$ s.t.$(Q, A) \in Att$. In other words, an argument is acceptable with respect to set $\mathcal{S}$ if $\mathcal{S}$ can *defend* it. Set $\mathcal{S}$ defends an argument if it attacks all the attackers of the argument.

Admissibility: An admissible set $\mathcal{S}$ is a conflict-free set in which all arguments are acceptable with respect to $\mathcal{S}$.

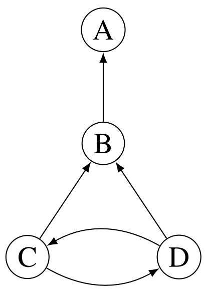
Complete Extension: A complete extension is an admissible extension, which includes all the acceptable arguments with respect to itself. That means, for a set $\mathcal{S}$ to be a complete extension, $\mathcal{S}$ should encompass all the arguments it can defend.

Grounded Extension: The grounded extension is the minimal (with respect to set inclusion) complete extension.

Preferred Extension: A preferred extension is the maximal (with respect to set inclusion) admissible extension.

Stable Extension: A stable extension is a complete extension that attacks all arguments that do not belong to it. Therefore, it includes all arguments that it can defend but it also attacks all those ones that is does not defend.

When an argument belongs to an extension, the argument is said to be acceptable with respect to that extension. Argumentation semantics, in more recent works, are defined based on a labellings system proposed by Caminada [2006]. These labellings provide an easy way to



Complete Extensions: $\{\}$,$\{A, C\}$, and $\{A, D\}$

Grounded Extensions: $\{\}$

Preferred Extensions: $\{A, C\}$, and $\{A, D\}$

Stable Extensions: $\{A, C\}$, and $\{A, D\}$

Figure 2-6: Argumentation Semantics

identify the status of arguments with respect to certain semantics. An argument is respectively, labelled in, out and $undec$, if it is acceptable, rejected and undecided under a certain semantics. Figure 2-6 illustrates a graphical representation of a DAF that is essentially a directed graph, in which arguments are represented by nodes and attacks are represented by arrows. The complete, grounded, preferred and stable extensions of the DAF displayed, are presented on the right hand side of the figure.

Dung's definition of an argumentation framework received a lot of attention and laid the foundation for many other frameworks such as *preference-based argumentation frameworks* [Amgoud and Cayrol, 2002], *value-based argumentation frameworks* [Dunne and Bench-Capon, 2004], *extended argumentation frameworks* [Modgil, 2007], *bipolar argumentation frameworks* Amgoud et al., 2004 and *assumption-based argumentation frameworks* [Bondarenko et al., 1993; Dung et al., 2009]. These frameworks attempt to address some issues that are not dealt with in DAF. For instance, DAF abstracts away the internal structure of arguments, which consequently does not allow defining *how* one argument attacks another one. The assumption is that the arguments and their interactions are given. Also DAF does not take into account the strength or weights of the arguments. In what follows, we give an overview of some influential argumentation frameworks that originated from Dung's and highlight how they extend DAF.

The preference-based argumentation framework (PAF) extends Dung's framework by defining a set of preferences over arguments to reflect the weight or importance of arguments. An attack from one argument to another one is only successful if the latter is not preferred over the former. A successful attack is referred to as a *defeat*. More discussion of PAF follows in Chapter 5 (page 104). Another development upon Dung's framework was introduced by Bench-Capon [2002] in the form of the value-based argumentation framework (VAF). Instead of preferences over arguments, VAF uses preferences over values to distinguish between attack and defeat: the attack of one argument to another one counts as defeat if the value of the latter is not preferred to the value of the former from a particular audience perspective. In VAF each argument can be mapped to different values by different audiences. An audience G is merely a total ordering on values from G's point of view. If we assume that V_1 and V_2 are both elements of V then G might prefer V_1 to V_2 and therefore ranks it higher than V_2, while audience H might do the reverse [Dunne and Bench-Capon, 2004]. Acceptability of arguments in this framework are thus said to be subjective. The same argument may convince audience G even though it clearly fails to convince audience H. More recently, Extended Argumentation Frameworks (ExAF) [Modgil, 2007] were introduced as an extension to Dung's framework. Unlike other frameworks, the attack relation in ExAF is not limited to an attack between arguments. An argument can attack an existing attack between two arguments as a way to express preferences and reduce symmetric attacks to asymmetric ones. Bipolar argumentation frameworks (BAF) [Amgoud et al., 2004] permit two different types of relation between arguments, namely defeat and support. By positive and negative relations, agents express their support for or against an argument, respectively. The Assumption-Based Argumentation Frameworks (ABA) [Dung et al., 2009] are another instance of DAF. However in contrast to DAF, ABA does not abstract away the internal structure of arguments. In this framework, arguments are deductions with assumptions as their premises. Since assumptions are open to challenge, an attack to an argument is an attack on its assumptions. The advantage of considering an internal structure for arguments lies in finding arguments and also the attack relations between them [Gaertner and Toni, 2007a] . Except for ABA, in all other frameworks mentioned in this section, this advantage is denied and arguments and the attack relations between them are presumed to be given. The internal structure of arguments is in particular important when argumentation is the tool with which an agent reasons about its beliefs and/or actions. In application-based domains, such as with agent reasoning, the assumption that arguments and their attacks are given is far too simplistic. Due to the importance of the internal structure of arguments in agent reasoning, we dedicate the next section to the internal structure of arguments.

Internal Structure of Arguments

Arguments represent defeasible logical inference and have been presented using *logic-based* – such as assumption-based [Bondarenko et al., 1993] – and *scheme-based* [Walton, 1996] approaches. When presented using logic, arguments are logical inferences from a set of premises to a set of conclusions [Amgoud and Prade, 2004a; Kraus et al., 1998; Prakken, 2010]. There are three types of attacks [Prakken, 2010] recognised between arguments: (i) rebuttal: when two arguments negate the conclusions of one another; (ii) undermine: when an argument negates the premises of another argument; and (iii) undercut: when one argument challenges the inference step in another argument. Scheme-based arguments, on the other hand, are based on argument schemes. Argument schemes are reasoning patterns expressed in natural language and critical questions are situations in which the schemes do not apply and are used to attack the arguments constructed based on the schemes. Argumentation schemes are especially popular in computational systems, when arguments need to be structured and formulated diversely so that they can capture the domain-dependent features of the problem they are modelling [Atkinson and Bench-Capon, 2007b; Toniolo, 2013; Walton, 1996]. The past decade has witnessed an increasing interest in the application of argumentation scheme in practical reasoning, planning and decision-making [Atkinson and Bench-Capon, 2007a; Atkinson et al., 2011; Gasque, 2013; Ouerdane et al., 2008; Toniolo et al., 2012]. In the remainder of this section, we discuss the background and origin of argument schemes as well as their applications in practical reasoning.

One of the earliest examples of the use of argument schemes is Toulmin's argument schema [Toulmin, 1958] that accounts for one of the most influential schemes in the field. This schema, as displayed in Figure 2-7, consists of six elements:

- Claim: a statement whose merits we are seeking to establish.
- Data: the fact we appeal to as a foundation for the claim.
- Warrant: the inference that takes us from data to the claim.
- Quantifier that indicates the strength of the warrant.
- Rebuttal: a condition under which the conclusion is defeated.
- Backing that represents the authority of the warrant.

Figure 2-8 shows an example of this schema with its six elements. Assume that Harry was born in Bermuda, on the account of statute "X", a man born in Bermuda will generally be a British citizen, so we can presume that Harry is a British citizen unless for example he has become a naturalised American.

Tolumin's schema, due to its expressivity and defeasible nature, has been implemented in a number of systems (e.g. [Bench-Capon and Staniford, 1995; Marshall, 1989]). However, its lack of certain features led to subsequent schemes, most popular of which are Walton's [Walton, 1996]. Atkinson, 2005 mentions the following as the reasons behind the shift from Toulmin's schema:(i) the schema does not clearly identify the manner in which an argument can be attacked; (ii) it is impossible to distinguish between different types of attacks such as rebuttal

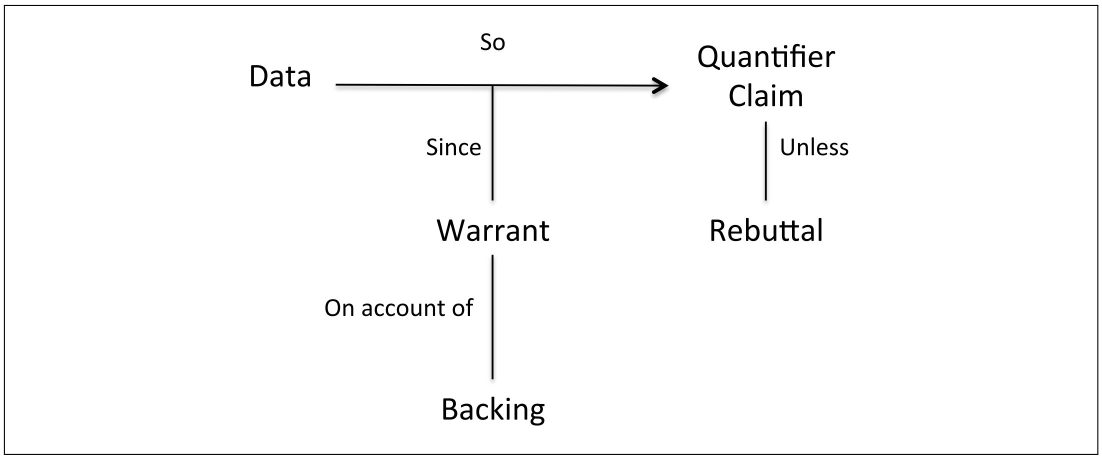


Figure 2-7: Toulmin's Argument Schema [Toulmin, 1958, p. 105]

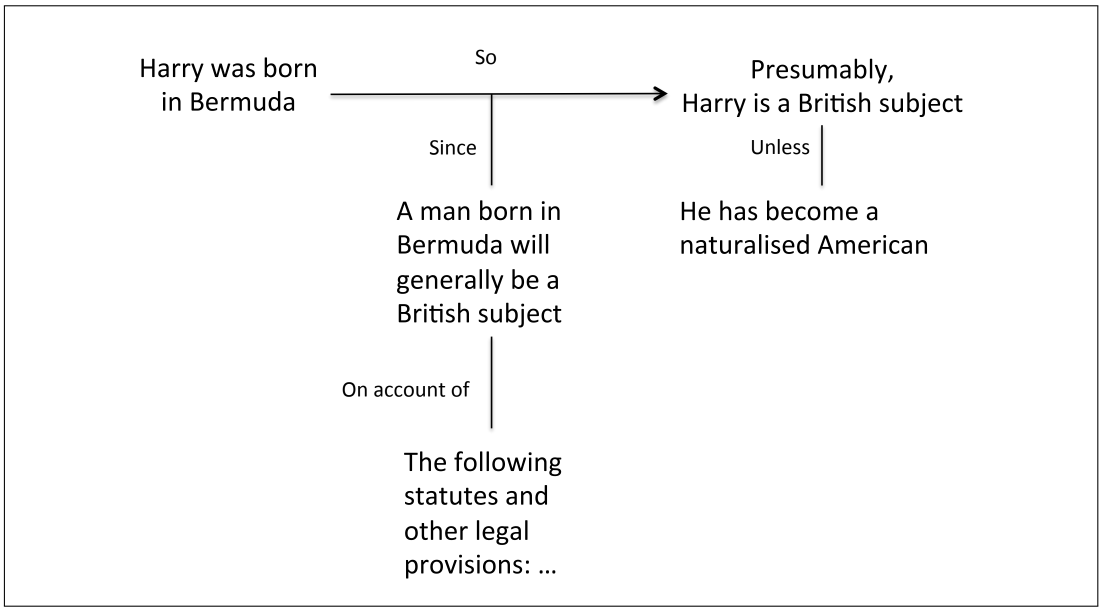

Harry was born in Bermuda
So
Presumably, Harry is a British subject
Since
Unless
A man born in Bermuda will generally be a British subject
He has become a naturalised American
On account of
The following statutes and other legal provisions: …

Figure 2-8: Toulmin's Argument Schema Example [Toulmin, 1958, p. 105]

(attack to the claim) and undercut[2] (attack to the support of the claim (i.e. data), or the inference resulted in the claim (i.e. warrant)); and (iii) there is no difference between arguments about beliefs and actions, despite the established differences between epistemic reasoning (reasoning about beliefs) and practical reasoning (reasoning about actions) (see Page 20).

In response to the shortcomings of Toulmin's schema, Walton introduced 26 argument schemes [3] [Walton, 1996] for various purposes. We mention few of these schemes by way of example:

Expert opinion scheme

- Source E is an expert in subject domain S containing proposition A.
- E asserts that proposition A is true (false).
- A is true (false).

Established rules scheme

- If A is the case, then an evaluation E is justified/ conduct C is required.
- A is the case.
- Therefore, evaluation E is justified/ conduct C is required.

Cause to effect scheme

- Generally, if A occurs, then B will (might) occur.
- In this case, A occurs (might occur).
- Therefore, in this case, B will (might) occur.

One of the schemes that Walton emphasises is the scheme for practical reasoning.

> "The analysis of argumentation schemes is very much affected by the recognition of practical reasoning as a distinctive type of reasoning, as distinguished from what might be called theoretical or discursive reasoning". [Walton, 1996, p. 11]

Walton's argument schemes for practical reasoning are based on two basic types of practical inferences, namely the *necessary condition scheme* and the *sufficient condition scheme* [Walton, 1996]. Both schemes are based on the idea that practical reasoning is reasoning toward goals

[2]Earlier in this section, following the reference [Prakken, 2010], we called this type of attack undermine. However this type of attack was previously called undercut, as is also the case in ABA [Dung et al., 2009].

[3]A more detailed classification of argument schemes can be found in Walton et al. [2008], where the authors present 60 categories of schema.

- G is a goal for a
- Doing A in necessary for a to carry out G
- Therefore, a ought to do A

- G is a goal for a
- Doing A in sufficient for a to carry out G
- Therefore, a ought to do A

CQ1: Are these alternative ways (other than A) of realising G?

CQ2: Is it possible for a to do A?

CQ3: Does a have goals other than G that should be taken into account?

CQ4: Are there other consequences of bringing about A that should be taken into account?

Figure 2-9: Walton's Practical Reasoning Schemes and Critical Questions [Walton, 1996, pp. 11-12]

conducted by an agent in a particular situation known by the agent. The conclusion of a practical reasoning scheme gives an agent an account of what to do in a given situation. However this conclusion can be challenged by four *critical questions*. The *necessary condition scheme* and the *sufficient condition scheme* along with their associated critical questions are presented in Figure 2-9. Figure 2-10 presents an example of each scheme.

Argument schemes for practical reasoning have proven to be greatly popular because they easily lend themselves to the defeasible nature of reasoning about action [Atkinson, 2005]. However, according to Atkinson [2005]; Atkinson and Bench-Capon [2007b], who have extensively explored the role of these schemes in practical reasoning since 2005, the notion of goals in Walton's argument schemes is "overloaded" and hence in need of further elaboration. They pinpoint the issue that goals in Walton's schemes encompass (i) the direct results of action; (ii) their consequences; and (iii) the reasons why those consequences are desired . Disambiguating the notion of goal by separating it into three elements (i), (ii), and (iii) results in Atkinson [2005] and Atkinson and Bench-Capon [2007b] argument scheme for practical reasoning. This scheme and its 16 critical questions are displayed in Figure 2-11. Atkinson shows

- I want to catch the train to London;
- Getting to the train station is necessary to catch the train;
- Therefore, I should run to the train station.

- I am thirsty;
- Drinking water is sufficient to remedy my thirst;
- Therefore, I should drink water.

Figure 2-10: Examples of Walton's Practical Reasoning Schemes

the application of this argument scheme in different domains such as eDemocracy, medicine and law [Atkinson, 2005]. The two examples in Figures 2-12 and 2-13 show the application of the scheme in eDemocracy and medicine, respectively. These examples and others that were presented in this section, show how the flexibility of argument schemes allows to define the internal structure of arguments to fit different purposes. Among other purposes, argument schemes for practical reasoning, have been exploited extensively. This section aimed mainly at covering scheme-based approaches to the internal structure of arguments involved in practical reasoning.

2.2.2 Argumentation for Agent Dialogue

As discussed in the previous section, argumentation has served as a computational mechanism for agent reasoning, which is often defeasible. To prove a defeasible claim one has to seek for any evidence to the contrary of the claim. The absence of such evidence is the proof itself. If such evidence is present, on the other hand, it will be treated as a new claim and therefore any evidence to contrary of it will be sought for and so on. This process refers to the dialogical aspect of argumentation and traces back to the work of Hamblin [1970, 1971]. He describes dialectical systems as regulated dialogues conducted by number of participants that take turn in making utterances in accordance with a set of rules. Utterances are often known as *moves* or *locutions* and the rules that regulate the moves are known as the dialogue *protocol*. To ensure the consistency of the utterances, participants need to keep a store of previous uttered statements representing their commitments. These stores are called *commitment stores* and can be modified according to a set of commitment rules defining the effect of moves on the

- In the current circumstances R
- We should perform action A
- Which will result in new circumstances S
- Which will realise goal G
- Which will promote value V

CQ1: Are the believed circumstances true?

CQ2: Assuming the circumstances, does the action have the stated consequences?

CQ3: Assuming the circumstances and that the action has the stated consequences, 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?

Figure 2-11: Atkinson's Practical Reasoning Scheme and Critical Questions [Atkinson and Bench-Capon, 2007b]

- Saddam is running an oppressive regime.
- we should invade Iraq
- to depose Saddam
- which will bring democracy to Iraq
- which will promote human rights.

Figure 2-12: Application of Atkinson's Scheme in eDemocracy [Atkinson, 2005, p. 144]

- Where there is a history of gastritis and no acid reducing therapy
- we should not prescribe aspirin
- so as not to cause excess acidity
- so as not to risk ulceration
- and so promote the value of safety.

Figure 2-13: Application of Atkinson's Scheme in Medicine [Atkinson, 2005, p. 158]

commitment store. Hamblin's work was continued by Mackenzie [1979, 1990] who developed four dialogue systems in the tradition of Hamblin, that were intended to identify and describe the properties of real-time argument games. Although intended for real-time and hence closer to real-life dialogues, Mackenzie makes it clear that none of these dialogues are yet adequate for any type of real-life argumentative dialogues [Mackenzie, 1990]. But what are real-life argumentative dialogues? This question was later on discussed by Walton and Krabbe [1995], who, according to the purpose of the dialogue, identified six classes of dialogues used in human communication:

Persuasion Dialogue: In this type of dialogue one agent tries to convince another agent to accept a viewpoint that the former holds, while the latter does not. Persuasion dialogue has been mainly used in the context of epistemic reasoning, examples of which can be found in Bentahar et al. [2004]; Caminada and Podlaszewski [2012b]; Devereux and Reed [2009]; Prakken [2006b].

Negotiation Dialogue: This type of dialogue is used when agents engage in a dialogue to find a way to allocate some scarce resource in a way that it is acceptable to all agents involved in the dialogue. Examples of negotiation dialogue in multi-agent systems appear in Amgoud and Vesic [2012]; Kraus et al. [1998]; McBurney et al. [2003]; Rahwan et al. [2003].

Eristic Dialogue: In this type of dialogue participants quarrel verbally with the aim of winning the exchange going on at any cost. This type of dialogue is not studied in agents and multi-agents systems as such.

Inquiry Dialogue: Agents involved in this type of dialogue collaborate to establish the truth value of a proposition whose value is not apparent to any of the parties. There are only a few examples of this type of dialogue present in the literature, e.g. Fan and Toni [2012]; Riley et al. [2011].

Deliberation Dialogue: During a deliberation dialogue, agents collaborate in order to decide what actions to take in a specific situation. This dialogue has been widely studied in multi-agent systems [Gasque, 2013; Kok et al., 2012; Tang and Parsons, 2005; Toniolo, 2013].

Information-Seeking Dialogue: In information-seeking dialogue an agent tries to discover the answer to a question from another agent that is believed by the first agent to know the answer. An example for this type of dialogue in the context of multi-agent systems is available in Fan and Toni [2012].

The six type of argumentative dialogues mentioned above make the assumption that the dialogue takes place between at least two agents for the purpose of reaching an agreement, or establishing the truth of a statement, etc. Moreover and however, a dialogue can be thought of taking place "in the mind" of a single agent in Gaertner's words [Gaertner, 2008, p. 13], in which case it is an internal dialogue contributing to the agent's reasoning process. Engaging in an internal dialogue to reason and act based on the outcome of reasoning was first introduced by Pollock [1995]. The development of this type of dialogue is greatly motivated by applications of argumentation in agent reasoning and decision-making [Vreeswijk and Prakken, 2000]. Next section includes more details on this type of dialogue and their applications in the mentioned areas.

2.2.3 Argumentation for Explanation

Explanation plays an important role in making artificial reasoning understandable and thus reliable for human users [Lacave and Díez, 2004; Wooley, 1998]. In the previous two sections we discussed the role of argumentation in agents reasoning and multi-agent dialogue. In addition to these roles, argumentation can serve as a tool for generating explanation [Baroni and Giacomin, 2009; Caminada et al., 2014c; Fan and Toni, 2015; García et al., 2013; Lacave and Díez, 2004; Schulz and Toni, 2014]. Intelligent agents equipped with argumentation capabilities can explain the validity of their recommendation to their users in a form of explanatory dialogues that are similar to human argumentation activities [Moulin et al., 2002]. These dialogues are referred to as dialogue games – also referred to as "argument games" or "proof theories". The dialectical explanation formalised through dialogue games relies on argumentation semantics. However, in contrast to semantics that justify the validity of an argument in terms of membership of a set, these dialogues provide dialectical explanation for arguments. Essentially, the aim of proof dialogue games is to create a link between argumentation as the basis of non-monotonic inference and argumentation in dialogue theory [Caminada, 2008; Caminada and Podlaszewski, 2012b].

The concept of argumentation frameworks and the role of argumentation semantics in serving as the basis of non-monotonic reasoning were discussed in Section 2.2.1. We recall here that argumentation as a mean for non-monotonic reasoning aids agents to (i) build arguments; (ii) identify their relationships; and (iii) evaluate the constructed argumentation framework based on argumentation semantics. The result of this evaluation creates a standpoint, based on which the agent can decide what arguments are acceptable. The idea in making a connection between argumentation as a mean for non-monotonic reasoning and argumentation as a dialectical process, is to establish whether an argument is accepted under certain semantics if it can be defended in a particular type of proof dialogue. Cayrol et al. [2001] define a proof dialogue as the combination of a dialogue type and a winning criterion that determines the winner of the dialogue. These proof dialogues are in the form of a dialectical argument game between a defender/proponent and a challenger/opponent. The game starts with an argument from proponent that needs to be tested. After that each of the players take turns in attacking other parties arguments with a counterargument. The initial argument in the game is acceptable if the proponent has a wining strategy and not acceptable otherwise. The winning strategy and rules of the argument game are defined based on the semantics for which the game is designed. Consequently, proof dialogues for different semantics are essentially dialogue games that tests if an argument put forward by a proponent is in extensions of that semantics.

In general, the preferred and grounded semantics among other semantics have received the most attention for being modelled as dialogues. Prakken [2006a] relates this observation with the fact that these two semantics are the only two semantics with "*elegant proof-procedures*" in argument game form. In what follows we briefly survey dialogue games.

Vreeswijk and Prakken [2000] were the first to present dialectical proof theories for arguments accepted under the preferred semantics. A slightly different version of their proof theory for the preferred semantics was proposed by Cayrol et al. [2001]. Cayrol et al. claim that using their proof theories, a proof given for an argument is *usually* shorter than when using the proof theories by Vreeswijk and Prakken [2000]. Modgil and Caminada [2009] specify argument games for preferred semantics based on Caminada's labellings [Caminada, 2006] that were mentioned earlier (page 34). Their argument games are similar to the ones defined in Vreeswijk and Prakken [2000], but as mentioned, differ from [Vreeswijk and Prakken, 2000], because the games are based on Caminada's labellings [Caminada, 2006]. Caminada [2010] makes a connection between the preferred semantics and the *Socratic discussion*. He believes that semantics that were originally defined by Dung [1995] are merely technical and mathematical definitions, making it difficult to grasp the reasoning concept behind the semantics. So modelling the preferred semantics as Socratic discussion is an attempt to bridge the gap between the mathematical formulation and the philosophical intuition behind the semantics. More on the connection between preferred semantics and the Socratic discussion follows in Chapter 6. Similarly, in [Caminada and Podlaszewski, 2012b], Caminada has formulated the grounded semantics in terms of a persuasion dialogue. Furthermore, Prakken [2006a] has proposed a dialectical proof theory that is a combination of dialogue games for the grounded and preferred semantics.

As mentioned earlier, the development of dialogue games is greatly motivated by applications of argumentation in agent systems [Vreeswijk and Prakken, 2000]. Dialogue games although formal, are known to be natural enough to be applicable in agent-to-agent and agent-to-human settings [Barbini et al., 2009; Caminada and Podlaszewski, 2012b; Caminada et al., 2014c; Vreeswijk and Prakken, 2000]. However, despite the main motivation behind the development of these types of dialogue games, they have rarely been used in agent-to-agent or agent-to-human settings. Two existing applications of dialogue games are [Zhong et al., 2014] and [Caminada et al., 2014c] that use the admissible and grounded semantics, respectively. In the former the authors use admissible dispute trees developed for Assumption-based Argumentation [Dung et al., 2009] to provide natural language explanation for why a certain decision is better than another one in a legal scenario. In Caminada et al. [2014c] a dialogical proof procedure based on the grounded semantics dialogue game [Caminada and Podlaszewski, 2012b] is created to justify the actions executed in a plan. The justification is mainly focused on the preconditions and effects of actions with regards to the goal state.

As pinpointed earlier in Section 2.1, our research is concentrated on practical reasoning. Despite the importance, the subject of explanation has received little attention in existing approaches to practical reasoning [Atkinson and Bench-Capon, 2007b; Broersen et al., 2001; Criado et al., 2010; Hulstijn and van der Torre, 2004; Kollingbaum and Norman, 2003; Rahwan and Amgoud, 2006; Sadri et al., 2006]. We have therefore used the advances made in dialogue games for the preferred semantics to propose a novel argumentation-based model that not only allows the agent to decide what plans to act upon, but also to benefit from the use of a structured dialogue to explain this decision. Why preferred extension should serve as the basis of practical reasoning is a question that we answer elaborately in Chapter 5 (page 113). The next section brings together the current section and the first section (Agent Reasoning) of this chapter to discuss the role of argumentation in practical reasoning and to survey the argumentation-based approaches to practical reasoning.

2.3 Argumentation-Based Practical Reasoning

In this section we provide a survey of approaches that use argumentation techniques for practical reasoning purposes. These approaches are either based on the Belief-Desire-Intention (BDI) architecture [Rao and Georgeff, 1995] or Action-based Alternating Transition System (AATS) [Hoek et al., 2007]. When modelling single agent practical reasoning process, BDI has been the most commonly used architecture to represent agents. In this architecture, beliefs represent the information that the agent has about the world, desires represent the agent's objectives and intentions represent the course of actions that given the current situation of the agent, can be taken to achieve particular desires. On the other hand, AATSs are widely used to represent all possible evolutions of a system due to the joint actions of multiple agents within it. Each action in this model has a set of preconditions that has to hold for the agent to be able to execute the action in that state. The result of executing these actions then causes the transition in the system. Section 2.3.1 reviews the approaches based on BDI, followed by a survey of approaches using AATSs in Section 2.3.2. Section 2.4 highlights the advantages and shortcomings of both categories.

2.3.1 BDI-based Approaches

Amgoud [2003] uses argumentation frameworks to detect the conflict between a set of inconsistent desires. Resolving the detected conflict results in obtaining consistent sets of intentions from a conflicting set of desires. Later on, Rahwan and Amgoud [2006] extend this approach by generating the desires in the first place. In this approach, Amgoud and Rahwan consider three different Dung-style argumentation frameworks for arguing about beliefs, desires and intentions. Arguments about beliefs include judging their truth value or checking them against observations, whereas arguing about desires addresses the justification of their adoption in the first place. Arguing about intention, on the other hand is concerned with what is the best course of actions to achieve desires, based on the worth of those desires and resources required to achieve them. So decision-theoretic notions determine what intentions the agent should pursue. Continuing the work of Rahwan and Amgoud [2006], Amgoud et al. [2008a] spotted that reasoning about desires and intentions in two argumentation frameworks raises the problem of committing to some desires that may not be feasible at all (an issue touched upon earlier in Section 2.1.1). To remedy this problem, Amgoud et al. [2008a] propose a constrained argumentation system for practical reasoning that restricts the desires to those that are both justified and feasible. Such a set of desires hold in the current state of the environment and there exists at least one plan that achieves each individual desire. Unlike Rahwan and Amgoud [2006], in this work there is no mechanism to compare various sets of justified and feasible desires.

Hulstijn and van der Torre [2004] agree with Amgoud [2003]; Rahwan and Amgoud [2006] on the fundamental difference in conflicts between beliefs and conflicts between plans. However, they are not in favour of using different argumentation frameworks to capture these differences. Instead, they extract goals by reasoning forward from desires, followed by deriving plans for goals, using planning rules. Goals that have a plan associated with them, can be modelled as an argument consisting of a claim and its necessary support. Goal arguments form an argumentation framework for planning in which there is an attack between conflicting plans. Conflict between plans is restricted to conflict between actions in the plans, where the conflict between actions is encoded as integrity constraints. They then look for an extension of this framework that maximises the number of achieved desires as opposed to Rahwan and Amgoud [2006] which focuses on the utility of goals/desires, rather than their quantity.

Different from above approaches, Gaertner [2008] permits norms to be a part of the agent's deliberation process. He proposes a norm-oriented BDI architecture in which norms of the society are internalised and blindly followed by the agent. Norms in this work act as *bridge rules* that dictate the relationship between the agent's mental attitudes, namely beliefs, desires and intentions. The conflict between norms is addressed using a hybrid argumentation framework based on Dung's and assumption-based argumentation framework. The addressed conflicts are resolved using preference information. The hybrid argumentation framework is implemented as a platform-independent system named CaSAPI [Gaertner and Toni, 2007a].

2.3.2 AATS-based Approaches

Atkinson and Bench-Capon [2007b] consider practical reasoning as a type of presumptive argumentation process to reason about what actions to perform. However, this presumption can be challenged by an argument scheme and associated critical questions (see Figure 2-11, page 42, for Atkinson's argument scheme and critical questions). In order to decide what to do, the agent's initial situation, alternative available actions from that situation and values promoted by those actions are instantiated within an AATS [Hoek et al., 2007]. Using this AATS, a set of arguments are generated for each available action. These arguments are then organised in a value-based argumentation framework (see Section 2.2.1), where the preference between arguments is defined according to the values they promote and the goals they contribute to. That said, value promotion here is treated qualitatively because there is no measurement of how much a value is promoted. The first limitation of the approach proposed in Atkinson and Bench-Capon [2007b], namely, *inexpressive* representation of goals is resolved in Atkinson and Bench-Capon [2014a]. The second limitation, which is the restricted consideration of an action's consequence for the immediate next state, is dealt with in Atkinson and Bench-Capon [2014b].

Oren [2013] proposes a normative practical reasoning framework based on AATS and Atkinson's argumentation scheme for practical reasoning [Atkinson, 2005]. This framework adopts several ideas from Atkinson and Bench-Capon [2007b], however, unlike Atkinson and Bench-Capon [2007b], it permits practical reasoning in the presence of norms. Moreover, different from Atkinson and Bench-Capon [2007b], Oren [2013] builds arguments for sequences of actions (i.e. paths) rather than individual actions. As a result, his schemes (see Figure 2-14) are much simpler than Atkinson's. Based on his scheme paths are mutually exclusive and the most preferred one is the one that has to be executed. The preferences between paths are defined based on considering all possible interactions between norms and goals instead of values and goals as it is in Atkinson and Bench-Capon [2007b]. The arguments, built based on the schemes and their interactions based on the critical questions, instantiate an extended argumentation framework (see Section 2.2.1) that is evaluated by applying the preferred semantics (see page 34).

Another work to note is Toniolo et al. [2012]. This approach uses argument schemes for collaborative planning in a normative environment. However the planning problem is not modelled in AATS, it is modelled in Situation Calculus (SC) [Reiter, 1991]. The argument scheme for norms deals with violation of norms as presented in Figure 2-15. As it is evident in the scheme, unlike Oren [2013], norms in Toniolo et al. [2012] are simply regimented, limiting the agent's normative reasoning capability to complying always with the imposed norms, without considering the possibility of violation. Permitting violation, allows the agent to weigh up outcomes of disregarding or adhering to a norm prior to committing to compliance or violation.

A summary of approaches discussed in this section and the previous section is provided in Table 2.1. The column labelled "Foundation" denotes the underlying architecture in each approach. The "Social" column shows the consideration of social aspects of agents, which is not available in the first two groups of approaches. In the third group the social aspects are

AS1: In situation S, the sequence of joint actions $A_1, \cdots, A_n$ should be executed.

- CQ1-1 Does some other sequence of actions exist that can be executed?
- CQ1-2 Is there a more preferred sequence of actions to this one?

AS2: The sequence of joint actions $A_1, \cdots A_n$ is preferred over $A'_1, \cdots A'_n$ as the former achieves a goal which the latter does not.

- CQ2-1 Is there some other sequence of actions which achieves a more preferred goal than the one achieved by this action sequence?
- CQ2-2 Does the sequence of actions lead to the violation of a norm?

AS3: The sequence of actions $A_1, \cdots, A_n$ should be less preferred than sequence $A'_1, \cdots A'_n$ as, in the absence of permissions, the former violates a norm while the latter does not.

- CQ3-1 Is the goal resulting from the sequence of actions more preferred than the violation?
- CQ3-2 Does the violation resulting from this norm result in some other, more important violation not occurring?
- CQ3-3 Is there a permission that derogates the violation?

AS4: There is a permission that derogates the violation of an obligation.

AS5: Agent α prefers goal g over goal g'.

AS6: Agent α prefers achieving goal g to not violating n.

AS7: Agent α prefers not achieving goal g to violating n.

AS8: Agent α prefers violating n to violating n'.

AS9: Agent α prefers situation A to B.

Figure 2-14: Oren's Argument Schemes for Normative Practical Reasoning [Oren, 2013]

AS: If a norm premise holds and the norm forbids/obliges performing an action or bringing about a state between some period, then the agent should not/must perform that action or bring about that state.

CQ Is there any norm that regulates actions or states of the world?

Figure 2-15: Tonolio's Argument Scheme for Norms

	Foundation	Sociality	AF
Amgoud, 2003, Rahwan and Amgoud, 2006, Amgoud et al., 2008a	BDI	N/A	DAF
Hulstijn and van der Torre, 2004	BDI	N/A	DAF
Gaertner, 2008	BDI	Norm	ABA
Atkinson and Bench-Capon, 2007b Atkinson and Bench-Capon, 2014b Atkinson and Bench-Capon, 2014a	AATS	Value	VAF
Oren, 2013	AATS	Norm	ExAF
Toniolo et al., 2012	SC	Norm	BAF

Table 2.1: Argumentation-based Frameworks for Practical Reasoning

taken into account in terms of what an agent values and cares about. For instance, the two values in the examples displayed in Figures 2-12 and 2-13, are "human rights" and "safety". To allow the representation of the values of the environment the agent operates in, the last two approaches integrates norms into agent's practical reasoning process. Note that values are what the agent internally cares about, whereas norms in [Oren, 2013] are environmental values and therefore external regulations imposed to the agent. Finally the last column illustrates what argumentation framework is used in each approach.

2.4 Summary

In this chapter we looked at the literature addressing practical reasoning for agents. We discussed how important the role of norm is in practical reasoning and what values and complexities it adds to an agent's practical reasoning. A number of solutions and approaches to normative practical reasoning are surveyed in Section 2.1.1, namely BOID [Broersen et al., 2001], NoA [Kollingbaum, 2005], López et al., 2005 framework, normative KGP [Sadri et al., 2006], Oren et al. [2011] and Panagiotidi et al. [2012b] approaches. Apart from these approaches, an extensive review of the argumentation-based approaches to practical reasoning is provided in Section 2.3. The reasons justifying the increasing popularity of argumentation techniques in practical reasoning are highlighted in Section 2.2 as (i) being able to deal with defeasible reasoning; and (ii) presentable in the form of a dialogue.

In short, we investigate the two main argumentation-based approaches to practical reasoning: logic-based approaches based on BDI and scheme-based approaches based on AATS. The majority of approaches using BDI [Amgoud, 2003; Amgoud et al., 2008a; Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006] are mainly focused on identifying a subset of consistent desires and their plans. However, it is not clear how, i.e. when and in which order, the agent should execute those plans, let alone the concurrency and interleaving aspects of planning. Bench-Capon and Atkinson [2009] point out that in these approaches, the focus is all on states and it is often difficult to distinguish between states and actions. A plan to achieve a desire is simply a start state and an end state in which the mentioned desire is achieved, without explicit representation of the actions that transformed the start state to the end state. As a result the intrinsic worth of actions, as the main component in any practical reasoning problem, is neglected. Another point that is also mentioned by Bench-Capon and Atkinson, is the absence of temporal aspects of practical reasoning in BDI-based approaches. In the second group of approaches, in order to develop a pattern of arguments to reason about action, a set of argument schemes and critical questions are employed. The flexibility of the schemes have made them popular in different domains of multi-agent systems such as practical reasoning [Atkinson and Bench-Capon, 2007b], planning [Gasque, 2013; Toniolo et al., 2012], normative reasoning [Oren, 2013], reasoning about trust [Parsons et al., 2014a]. For practical reasoning purposes, Atkinson and Bench-Capon [2007b] uses an argument scheme as presented in Figure 2-11, to justify actions. Unlike BDI approaches, in this approach, due to use of AATS, there is an explicit representation of actions as transitions between states, but states retain their primacy. Time representation, although present, is restricted to the single next step, and not capable of being explicitly expanded into a whole sequence of actions. Value promotion or demotion that is the central element in evaluating which actions to take, are also limited to the immediate consequence of action in the next state. Despite the more mature handling of state, action and time, we believe Pollock's criticism (see page 23) to approaches that apply decision theoretic concepts to actions rather than plans, stands for this approach too. Values in this approach, in essence, act as a measurement just like utilities. Therefore, according to Pollock, one has to check how a value is promoted or demoted throughout the whole course of action rather than a step-by-step treatment of value promotion or demotion for each single action.

In this thesis we propose an argumentation-based approach to practical reasoning that exploits both features of argumentation in handling defeasible reasoning and presenting the process in a form of a dialogue. In contrast to existing approaches, the approach we propose:

1. Does not abstract away the planning aspect of practical reasoning problem and hence limit the plans to a start and an end state.
2. Develops a formal model for normative practical reasoning that:
    - Clearly distinguishes between actions and states;
    - Handles time explicitly; and
    - Allows durative actions to be executed concurrently.
3. Applies argumentation techniques to plans, in deciding which plan to pursue, rather than to actions in a plan.
4. Uses an internal dialogue game to create transparency and explain why a certain plan should be executed, given an agent with some goals and norms.
5. Provides a translation of the dialogue game into natural language for ease of use for human users.

In the next chapter we present a model that defines the plans. The implementation of the model in Chapter 4 generates the plan. Plans are reasoned about in Chapter 5, using argumentation frameworks. The evaluation of argumentation frameworks aims at recognising the best plan(s) for the agent to execute. The explanation of the best plan in natural language using dialogue games is provided in Chapter 6.

Chapter 3

A Model for Normative Practical Reasoning

This chapter introduces a formal model and its semantics for normative practical reasoning. There are many different planning and action languages available, such as STRIPS [Fikes and Nilsson, 1971], event calculus [Kowalski and Sergot, 1986], Planning Domain Definition Language (PDDL) and its extensions [Fox and Long, 2003; Mcdermott et al., 1998], Temporal Action Logics (TAL) [Doherty et al., 1998] and Action Description Language (ADL) [Pednault, 1994]. STRIPS is the most well-established planning domain language that is the foundation of many automated planning languages such as PDDL and ADL. In this work, we need and action language that allows representing agent's actions with pre and postconditions. These actions are shaped by agent's goals and norms and the interactions between goals and norms. Also the agent should be capable of reasoning about temporal aspect of actions. We selected STRIPS as the foundation of our normative practical reasoning model, since although it is not the most expressive action language, it is expressive enough to represent temporal actions with pre and postconditions. In addition, it is flexible enough to be extended such that it caters for norms as well as goals.

In STRIPS a planning problem is defined in terms of an initial state, a goal state and a set of operators (e.g. actions). Each operator has a set of preconditions that denote the conditions under which the operator can be executed, and a set of postconditions that result from applying the operator. Any sequence of actions that satisfies the goal is a solution to the planning problem. In order to capture the features of the normative practical reasoning problem we are going to model, in section 3.1 we extend the classical planning problem by:

(i) replacing atomic actions with *durative* actions: often the *nature* of the actions is non-atomic, which means that although executed atomically in a state, the system state in which they finish executing is not necessarily the same in which they started [Nunes et al., 1997]. Refinement of atomic actions to durative actions reflects the real time that a machine takes to execute certain actions, which is also known as “real-time duration” of actions [Börger and Stärk, 2003].

(ii) Allowing a set of potentially inconsistent goals instead of the conventional single goal: the issue of planning for multiple goals distributed across multiple agents, when the agents share resources or when they need to collaborate to satisfy a common goal, is addressed in collaborative planning. However, the issue of a single agent trying to plan for multiple goals that are not necessarily consistent has not received much attention from a planning perspective. That said, multiple desires that are not consistent are common in BDI agents, but such agents are often provided with a plan library from which they can choose multiple plans that can satisfy multiple goals. However, the agent is assumed to be executing those plans in sequence and the interleaving of plans is not discussed. We will address the issue of plan interleaving for a single agent when handling multiple conflicting goals.

(iii) Adding a set of norms: having made a case for the importance of norms in practical reasoning in the previous chapter, we will now integrate normative and practical reasoning. Just like goals, a set of norms is not necessarily consistent, making it potentially impossible for the agent to comply with all norms imposed on it.

In addition to inconsistency within the goal set and the norm set, the set of goals and norms can also exhibit inconsistency. Therefore, defining a solution for such a planning problem requires considering all variations of conflict that may arise between these entities. In general, a solution for a planning problem that features (i), (ii) and (iii) above is any sequence of actions that satisfies at least one goal, while remaining conflict free. A sequence of action is conflict free if it does not include conflicting actions, does not satisfy conflicting goals, does not comply with conflicting norms, and does not satisfy and comply with conflicting goals and norms. The model allows the sequence of actions to be executed *concurrently*. The agent has the choice of violating or complying with norms triggered by execution of a sequence of actions, while satisfying its goals. However, there may be consequences either way that the agent has to foresee. The syntax and semantics of the model are explained in Sections 3.1 and 3.2, followed by the summary of this chapter in Section 3.3.

3.1 Syntax

As mentioned earlier, the foundation of the model we propose in this section is standard STRIPS-style planning. We therefore start this section by describing the syntax for STRIPS planning and then extend it, such that it can accommodate the following features (i) durative actions; (ii) multiple goals and (iii) multiple norms.

Definition 1 (STRIPS Planning [Fikes and Nilsson, 1971])**.** *A STRIPS planning problem is a tuple of form* $P = (\Delta, g, A)$*, where*

- Δ *is the initial state, which is a set of well-formed formulas.*
- g *is the goal state expressed as a well-formed formula.*
- A *is a set of actions that are each defined by an action description consisting of two main parts: the conditions under which the action is applicable and the effects of the action defined by a list of literals that must be added to the state and a list of literals that are no longer true and therefore must be deleted.*

Definition 2 (Normative Planning Problem)**.** *A normative planning problem is a tuple* $P = (FL, \Delta, A, G, N)$ *where*

- FL *is a set of fluents;*
- Δ *is the initial state;*
- A *is a finite, non-empty set of durative STRIPS-like [Fikes and Nilsson, 1971] actions for the agent;*
- G *denotes the set of agent goals;*
- N *denotes a set of norms imposed on the agent that define what an agent is obliged or forbidden to do under certain conditions.*

We now describe each of these items in more detail. The last three items, namely actions, goals and norms are each given a separate section due to their importance.

Fluents

FL is a set of domain fluents that accounts for the description of the domain the agent operates in. A literal l is a fluent or its negation i.e. $l = \mathit{fl}$ or $l = \neg \mathit{fl}$ for some $\mathit{fl} \in FL$. For a set of literals L, we define $L^+ = \{\mathit{fl} | \mathit{fl} \in L\}$ and $L^- = \{\mathit{fl} | \neg \mathit{fl} \in L\}$ to denote the set of positive and negative fluents in L, respectively. L is well-defined if there exists no fluent $fl \in FL$ such that $fl \in L$ and $\neg fl \in L$, i.e. if $L^+ \cap L^- = \emptyset$.

The semantics of the normative planning problem are defined over a set of states Σ. A state $s \subseteq FL$ is determined by set of fluents that hold *true* at a given time, while the other fluents (those that are not present) are considered to be false. A state $s \in \Sigma$ satisfies fluent $fl \in FL$, denoted $s \models fl$, if $fl \in s$. It satisfies its negation $s \models \neg fl$ if $fl \notin s$. This notation can be extended to a set of literals as follows, set X is satisfied in state s, $s \models X$, when $\forall x \in X \cdot s \models x$.

Initial State

The set of fluents that hold at the initial state is denoted by $\Delta \subseteq FL$.

3.1.1 Actions

A is a set of durative STRIPS-like actions, that is actions with preconditions and postconditions that take a non-zero duration of time to have their effects in terms of their postconditions.

Definition 3 (Durative Action)**.** *A durative action $a = \langle pr, ps, d\rangle$ is composed of well-defined sets of literals $pr(a), ps(a)$ that represent a's preconditions and postconditions and a positive number $d(a) \in \mathbb{N}$ for its duration. Postconditions are further divided into a set of add postconditions $ps(a)^+$ (positive literals in $ps(a)$) and a set of delete postconditions $ps(a)^-$ (negative literals in $ps(a)$).*

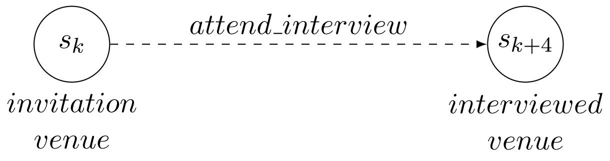
An action a can be executed in a state s if its preconditions hold in that state (i.e. $s \models pr(a)$). When modelling atomic actions, the system state in which the action execution starts and the state in which the action ends are the same. In contrast, when modelling durative actions, there might be several states between the start and end state of the action, during which the action is said to be *in progress*. Some approaches take the view that it is sufficient for the preconditions of the action to hold at the start state and it does not matter whether they hold while the action is in progress [Knoblock, 1994]. Whereas, some others hold that the preconditions of action should be preserved while the action is in progress [Blum and Furst, 1997]. Moreover, some planning languages, such as Planning Domain Description Language (PDDL) [Fox and Long, 2003; Garrido et al., 2002], distinguish between preconditions and those conditions that have to hold while the action is in progress. The latter conditions are referred to as *invariant conditions*. Having invariant conditions different from preconditions, undoubtedly, brings more expressiveness to the planning language, however it comes at the price of higher implementation complexity. In this research, we take the position that the invariant conditions are the same as preconditions, which implies that the preconditions have to be preserved throughout the execution of the action.

The postconditions of a durative action are applied in the state s at which the action ends, by adding the positive postconditions belonging to $ps(a)^+$ to s and deleting the negative postconditions belonging to $ps(a)^-$ from s. As a result we have: $s \models ps(a)^+$ and $s \not\models ps(a)^-$. Additionally, actions can be executed concurrently iff:

(i) They do not start at the same time: This is because the planning problem is defined for a single agent and a single agent is not typically assumed to be able to start two actions at the exact same instant.

(ii) They do not have concurrency conflicts, where concurrency conflict is caused if the pre or postconditions of an action is logically inconsistent with the pre or postconditions of another action (see next section for the formal definition of concurrency conflict).

Example 3.1. *Assume that "$attend\_interview$" is one of the actions available to the agent that takes 4 units of time. To attend the interview the agent has to have an invitation letter and to be present at the venue, which results in the agent being admitted to interview and consequently "interviewed". Once the interview is done, the invitation cannot be used for a second time:* $attend\_interview = \langle\{invitation, venue\}, \{interviewed, \neg invitation\}, 4\rangle$.



Concurrency Conflict

Actions can experience different types of conflict including constant and temporary. When two actions are in constant conflict, the agent cannot possibly execute both of them. On the other hand, a temporary conflict prevents the agent from executing two conflicting actions under specific constraints, the most common one of which is time. Conflict caused by time, known as a *concurrency conflict* between actions, prevents them from being executed in an overlapping period of time. Blum and Furst [1997] define that two actions a_i and a_j cannot be executed concurrently, if at least one of the following holds:

1. The preconditions of a_i and a_j contradict each other:
$\exists r \in pr(a_i)$ s.t. $\neg r \in pr(a_j)$ or
$\exists \neg r \in pr(a_i)$ s.t. $r \in pr(a_j)$

2. The postconditions of a_i and a_j contradict each other:
$\exists r \in ps(a_i)^+$ s.t. $\neg r \in ps(a_j)^-$ or
$\exists \neg r \in ps(a_i)^-$ s.t. $r \in ps(a_j)^+$

3. The postconditions of a_i contradict the preconditions of a_j:
$\exists r \in ps(a_i)^+$ s.t. $\neg r \in pr(a_j)$ or
$\exists \neg r \in ps(a_i)^-$ s.t. $r \in pr(a_j)$

4. The preconditions of a_i are contradicted by the postconditions of a_j:
$\exists r \in pr(a_i)$ s.t. $\neg r \in ps(a_j)^-$ or
$\exists \neg r \in pr(a_i)$ s.t. $r \in ps(a_j)^+$

We summarise the four conditions above in Definition 4, where we define what are referred to as conflicting actions in the remainder of this work.

Definition 4 (Conflicting Actions)**.** *Actions a_i and a_j have a concurrency conflict iff the preconditions or postconditions of a_i contradict the preconditions or postconditions of a_j. The set of conflicting actions is denoted as cf_{action}:*

$$\begin{aligned} cf_{action} = \{(a_i, a_j) \textit{ s.t. } \exists r \in pr(a_i) \cup ps(a_i)^+, \neg r \in pr(a_j) \cup ps(a_j)^- \textit{ or} \\ \exists \neg r \in pr(a_i) \cup ps(a_i)^-, r \in pr(a_j) \cup ps(a_j)^+\} \end{aligned} \tag{3.1}$$

Example 3.2. *Assume that the agent wants to take the action of attending a course to get some certificate. Once the fee for the course is paid, the precondition for this action is met and the agent is able to attend the course which results in the course being attended and a certificate of attendance being received: $attend\_course = \langle \{fee\_paid\}, \{course\_attended, certificate\}, 3\rangle$. But if the agent does not have the money to pay for the course, (s)he can rely on the company's funding to pay the fee: $comp\_funding = \langle \{\neg fee\_paid, \neg money\}, \{fee\_paid\}, 4\rangle$. The preconditions of these two actions are inconsistent which prevent them from being executed concurrently, however action $comp\_funding$ effectively provides the precondition for $attend\_course$ which means they can indeed be executed consequently.*

3.1.2 Goals

Goals are the central issue in any planning or practical reasoning problem. They identify the state of affairs in the world that an agent wants to satisfy. Different types of goals and their characteristics have been identified in the literature [Riemsdijk et al., 2008]. Figure 3-1 displays a classification of goals in which goals are divided into two broad categories of *declarative* and *procedural* goals. The former category is concerned with the result of execution of

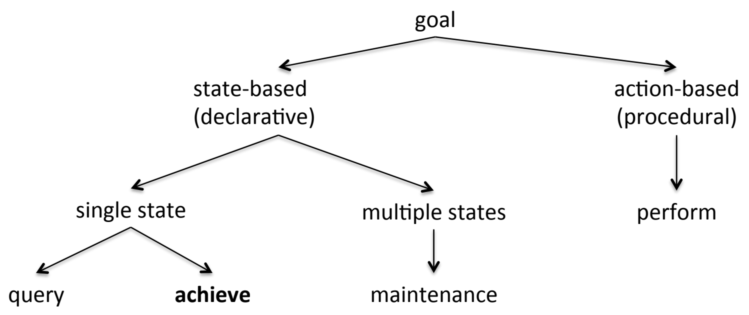


Figure 3-1: Goal Taxonomy [Riemsdijk et al., 2008]

actions, whereas the latter category is concerned with the actions themselves. Declarative or state-based goals are further divided into goals that need to achieve a certain state of affair (*achievement* goals) and goals that need to maintain a certain state of affairs (*maintenance* goals). *Query* goals, also referred to as test goals, are used to query the state of the agent about certain piece of information. *Achievement* goals are the most common type of goals modelled in the agent literature and have therefore received the most attention [Boer et al., 2002; Nigam and Leite, 2006; Riemsdijk et al., 2002, 2008].

Going back to our planing problem $P = (FL, \Delta, A, G, N)$, we now define G. Similar to [Boer et al., 2002; Nigam and Leite, 2006; Riemsdijk et al., 2002, 2008], goals for the purpose of this research are achievement goals. Thus, G denotes a set of (possibly inconsistent) achievement goals, each of which $g \in G$ is defined as a well-defined set of literals, known as goal requirements (denoted r_i), that should hold in order to satisfy the goal.

Definition 5 (Goal)**.** *Goal $g = \{r_1, \cdots, r_n\}$, where r_i is a literal, is satisfied in state s when $s \models g$.*

Example 3.3. *Let us assume an agent wants to join in a planned strike to support some union. In doing so, the agent has to be a member of the union, not go to the office and not attend any meeting on behalf of its company elsewhere. So we have:*

$$strike = \{union\_member, \neg office, \neg meeting\_attended\}$$

3.1.3 Norms

In Section 2.1.2 we explained what norms are and why they play an important role in agent practical reasoning. In this section we specify what we refer to as a norm in this thesis. In order to provide a context for the norm specification we propose, firstly, it is important to recall from the previous chapter (page 27) the five elements identified by Pacheco [2012] that distinguish norm specification languages. Secondly, we explain how our norm specification corresponds to those elements. The properties are:

1. Deontic Operators: in this work, we model a permissive society in which the agent has complete knowledge of the domain of actions available to it. Everything is permitted unless it is explicitly prohibited. The role of obligation is to motivate the agent to execute a specific action and the role of prohibition is to inhibit the agent from executing a particular action. The approach of modelling permission norms as exceptions to obligation and prohibition norms (see Chapter 2, page 26) is considered as part of future work.

2. Controls: controls determine whether the deontic propositions operate on actions, states or both. Existing approaches are divided into three categories depending on whether deontic operators control (i) states [Cliffe et al., 2005; Dastani et al., 2009; Fornara and Colombetti, 2007; Oren et al., 2008]; (ii) actions/events [Dignum, 2004; García-Camino et al., 2005; Hübner et al., 2007; Uszok et al., 2008]; (iii) states and actions [De Vos et al., 2013; Vázquez-Salceda et al., 2004] . So far in this thesis we have focused on action-based norms.

3. Enforcement Mechanisms: the enforcement mechanism we use is adopted from López et al. [2005], which is called "pressured norm compliance" and was explained in the previous chapter (page 31). In this method, what determines compliance is the conflict between goals and norms such that if there is a conflict between a norm and the agent's goals, the agent does not comply unless the goals hindered by punishment are more important than the goals facilitated by compliance. On the other hand, if there is no such conflict, the agent only complies with a norm if there are goals that are hindered through the punishment for a violation, and violates the norms otherwise. Similarly, we use the concept of conflict to persuade the agent to comply with a norm, however in contrast,

    - the conflict and hence the compliance mechanism is extended to cater for not only goal-norm conflict, but also norm-norm conflict. Therefore, when deciding whether to follow a norm that hinders a goal or another norm, the importance of the norm has to be weighed up against the hindered goal or norm; and
    - if there is no conflict between a norm and another norm or agent's goals, the norm has to be complied with.

4. Conditional Expressions: similar to the control element, we use actions as conditional expressions. In other words, the norm condition is an action that once executed, the agent is obliged to or prohibited from executing the action that is subject to control.

5. Temporal Constraints: temporal constraints can be used to express norm activation, termination, deadline, etc. The temporal constraint we specify here is concerned with the deadline. The agent is expected to comply with an obligation (i.e. execute a certain action) or a prohibition (refrain from executing a specific action) before some deadline. As well as temporal constraints, a deadline can be expressed as a state [1] [Pacheco, 2012]. However, it is straightforward to represent the norm deadline as a future time instant, rather than a state to be brought about. Associating a deadline with temporal properties is considered to be more realistic and more dynamic, in particular when the norms capture the requirements of real-world scenarios [Chesani et al., 2013; Gasparini et al., 2015; Kafali et al., 2014].

Having explained the properties of our norm specification language, we now define the element N of the planning problem $P = (FL, \Delta, A, G, N)$. N denotes a set of conditional norms to which the agent is subject:

Definition 6 (Norm)**.** *A norm is a tuple of the form* $n = \langle d\_o, a_{con}, a_{sub}, dl \rangle$*, where*

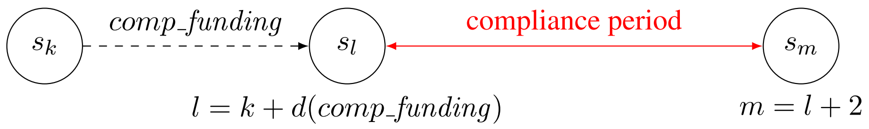
- $d\_o \in \{o, f\}$[2] *is the deontic operator determining the type of norm, which can be an obligation or a prohibition;*
- $a_{con} \in A$ *is the action that activates the norm;*
- $a_{sub} \in A$ *is the action that is the subject of the obligation or prohibition; and*
- $dl \in \mathbb{N}$ *is the norm deadline relative to the activation condition, which is the completion of the execution of the action* a_{con}*.*

An obligation norm expresses that taking action a_{con} obliges the agent to take action a_{sub} within dl time units of the end of execution of a_{con}. Such an obligation is complied with if the agent starts executing a_{sub} before the deadline and is violated otherwise. A prohibition norm expresses that taking action a_{con} prohibits the agent from taking action a_{sub} within dl time units of the end of execution of a_{con}. Such a prohibition is complied with if the agent does not start executing a_{sub} before the deadline and is violated otherwise.

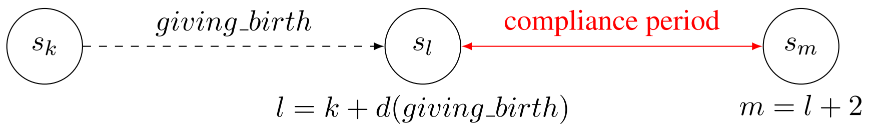
Example 3.4. *Assume that as an employee of a company, an agent is entitled to use company funding to attend some training, however, that obliges the agent to attend some meeting on behalf of the company before deadline 2:* $n = \langle o, comp\_funding, attend\_meeting, 2 \rangle$*.*

[1]When defined as state, deadlines are also referred to as a termination or expiration condition.

[2]o and f are normally denoted **O** and **F** in deontic literature. However we have used small letters to make it consistent with the implementation in the next chapter. Capital letters in the implementation language are reserved for variables.



Example 3.5. *Compulsory maternity leave prevents female employees to get back to work within two weeks of giving birth. This situation can be modelled as a prohibition norm that enforces the regulation:* $n = \langle f, giving\_birth, work, 2\rangle$.



3.2 Semantics

Having explained the syntax of the model, we now focus on the semantics. To this end, we first need to describe given a planning problem $P = (FL, \Delta, A, G, N)$:

(i) what the possible courses of action for the agent are and what properties each course of action has. Properties are defined in terms of the goals that a sequence of action satisfies, the norms it complies with and the norms it violates. This item is further discussed in Section 3.2.1.

(ii) What different type of conflicts the agent can experience while trying to satisfy its goals and comply with the norms to which it is subject. See Section 3.2.2 for more information on this item.

(iii) What identifies a sequence of actions as a solution/plan for problem P. Plans are defined in Section 3.2.3.

3.2.1 Sequences of Actions and their Properties

Let $P = (FL, \Delta, A, G, N)$ be a normative planning problem. Also let $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n})\rangle$ with $a_i \in A$ and $t_{a_i} \in \mathbb{Z}^+$ be a sequence of actions a_i executed at time t_{a_i}. The pair (a_i, t_{a_i}) reads as action a_i is executed at time $t_{a_i} \in \mathbb{Z}^+$ s.t. $\forall i < j, t_{a_i} < t_{a_j}$. The total duration of a sequence of actions, $Makespan(\pi)$, is given by Equation 3.2.

$$Makespan(\pi) = max(t_{a_i} + d(a_i)) \tag{3.2}$$

Definition 7 (Sequence of States)**.** *Let $\pi = \langle(a_0, 0), \cdots, (a_n, t_{a_n})\rangle$ be a sequence of actions such that $\nexists(a_i, t_{a_i}), (a_j, t_{a_j}) \in \pi$ s.t. $t_{a_i} \leq t_{a_j} < t_{a_i} + d(a_i), (a_i, a_j) \in cf_{action}$ and let $m = Makespan(\pi)$. The execution of a sequence of actions π from a given starting state $s_0 = \Delta$ brings about a sequence of states $S(\pi) = \langle s_0, \cdots s_m\rangle$ for every discrete time interval from 0 to m.*

The transition relation between two states is given by Definition 8. If an action a_i ends at time k, state s_k results from removing all delete postconditions and adding all add postconditions of action a_i to state s_{k-1}. If there is no action ending at s_k, the state s_k remains the same as s_{k-1}. We first define A_k as the set of action, time pairs such that the actions are ending at some specific state k:

$$A_k = \{(a_i, t_{a_i}) \in \pi \text{ s.t. } k = t_{a_i} + d(a_i)\} \tag{3.3}$$

Note that s_k is always well-defined since two actions with inconsistent postconditions, according to Definition 4 belong to cf_{action} and therefore cannot be executed concurrently and consequently cannot end at the same state: $\nexists(a_i, t_{a_i}), (a_j, t_{a_j}) \in \pi$ s.t. $(a_i, a_j) \in cf_{action}$.

Definition 8 (State Transition)**.** *Let $\pi = \langle(a_0, 0), \cdots, (a_n, t_{a_n})\rangle$ and let $S(\pi) = \langle s_0, \cdots s_m\rangle$ be the sequence of states brought about by π:*

$$\forall k > 0: \ s_k = \begin{cases} (s_{k-1} \setminus \bigcup\limits_{a \in A_k} ps(a)^-) \cup \bigcup\limits_{a \in A_k} ps(a_i)^+ & A_k \neq \emptyset \\ s_{k-1} & A_k = \emptyset \end{cases} \tag{3.4}$$

Definition 9 (Goal Satisfaction)**.** *A sequence of actions $\pi = \langle(a_0, 0), \cdots, (a_n, t_{a_n})\rangle$: satisfies goal g iff there is at least one state $s_k \in S(\pi)$ that satisfies the goal:*

$$\pi \models g \textit{ iff } \exists\, s_k \in S(\pi) \textit{ s.t. } s_k \models g \tag{3.5}$$

The set of goals satisfied by π is denoted as G_π:

$$G_\pi = \{g \text{ s.t. } \pi \models g\} \tag{3.6}$$

Definition 10 (Obligation Compliance)**.** *A sequence of actions $\pi = \langle(a_0, 0), \cdots, (a_n, t_{a_n})\rangle$ complies with obligation $n = \langle o, a_{con}, a_{sub}, dl\rangle$ iff the action that is the norm activation condition, a_{con}, has occurred and the action that is the subject of the obligation, a_{sub}, occurs* *between when the condition holds and when the deadline expires.*

$$\pi \models n \textit{ iff } (a_{con}, t_{a_{con}}), (a_{sub}, t_{a_{sub}}) \in \pi \textit{ s.t.} \\ t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), dl + t_{a_{con}} + d(a_{con})) \quad (3.7)$$

Definition 11 (Obligation Violation). *A sequence of actions* $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ *violates obligation* $n = \langle o, a_{con}, a_{sub}, dl \rangle$ *iff the action that is the norm activation condition,* a_{con}*, has occurred, but* a_{sub} *does not occur in the period between when the condition holds and when the deadline expires, the obligation is violated.*

$$\pi \not\models n \textit{ iff } (a_{con}, t_{a_{con}}) \in \pi, \nexists (a_{sub}, t_{a_{sub}}) \in \pi \textit{ s.t.} \\ t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), dl + t_{a_{con}} + d(a_{con})) \quad (3.8)$$

Definition 12 (Prohibition Compliance). *A sequence of actions* $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ *complies with prohibition* $n = \langle f, a_{con}, a_{sub}, dl \rangle$ *if the action that is the norm activation condition,* a_{con}*, has occurred and the action that is the subject of the prohibition,* a_{sub}*, does not occur in the period between when the condition holds and when the deadline expires.*

$$\pi \models n \textit{ iff } (a_{con}, t_{a_{con}}) \in \pi, \nexists (a_{sub}, t_{a_{sub}}) \in \pi \textit{ s.t.} \\ t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), dl + t_{a_{con}} + d(a_{con})) \quad (3.9)$$

Definition 13 (Prohibition Violation). *A sequence of actions* $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ *violates prohibition* $n = \langle f, a_{con}, a_{sub}, dl \rangle$ *if the action that is the norm activation condition,* a_{con}*, has occurred and* a_{sub} *occurs in the period between when the condition holds and when the deadline expires, the prohibition norm is violated.*

$$\pi \not\models n \textit{ iff } (a_{con}, t_{a_{con}}), (a_{sub}, t_{a_{sub}}) \in \pi \textit{ s.t.} \\ t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), dl + t_{a_{con}} + d(a_{con})) \quad (3.10)$$

Definition 14 (Activated Norms). *A norm* $n = \langle o|f, a_{con}, a_{sub}, dl \rangle$ *is activated in a a sequence of actions* $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ *if its activation condition* a_{con} *belongs to the sequence of actions. Let* N_π *be the set of activated norms in* π*:*

$$N_\pi = \{ n = \langle o|f, a_{con}, a_{sub}, dl \rangle \in N \textit{ s.t. } a_{con} \in \pi \} \quad (3.11)$$

The set of norms complied with and violated in π are denoted as $N_{cmp(\pi)}$ and $N_{vol(\pi)}$:

$$N_{cmp(\pi)} = \{n \in N_\pi \text{ s.t. } \pi \models n\} \tag{3.12}$$

$$N_{vol(\pi)} = \{n \in N_\pi \text{ s.t. } \pi \not\models n\} \tag{3.13}$$

To make sure there are no norms pending at $m = makespan(\pi)$, we assume that the norm deadlines are smaller than m. Therefore, all the activated norms in π are either complied with or violated by time m:

$$N_\pi = N_{cmp(\pi)} \cup N_{vol(\pi)} \tag{3.14}$$

3.2.2 Conflict

In this section we look at three types of conflict, namely, conflict between goals, conflict between norms and conflict between goals and norms. Conflict between goals and between goals and norms is a static property and does not depend on the sequence of actions. Conversely, conflict between actions is temporal and differs from one sequence of actions to another. Since norms are action-based, the same applies to conflict between norms. We explain static and temporal conflicts in more detail in the remainder of this section. There are examples of each type of conflict to demonstrate the concept in each case.

Goal-goal Conflict

An agent may pursue multiple goals or desires at the same time and it is very possible that some of these goals conflict [Nigam and Leite, 2006; Pokahr et al., 2005; Riemsdijk et al., 2002, 2009; Thangarajah et al., 2003]. Conflict between the agent's goals or desires, especially for BDI agents, has been addressed in several works. Hulstijn and van der Torre [2004] describe two goals as conflicting if achieving them requires taking two conflicting actions, where conflicting actions are encoded using integrity constraints. Rahwan and Amgoud [2006] on the other hand, define two desires as conflicting if the sets of beliefs that supports the achievement of desires are contradictory. Like Rahwan and Amgoud [2006], Broersen et al. [2002] argue that for a set of goals to not to be conflicting, a consistent mental attitude (e.g. beliefs and norms) is required. Some (e.g. Toniolo [2013]) have adopted a static view on goal conflict, in which conflicting goals are mutually-exclusive, hence impossible to satisfy in the same plan regardless of the order or choice of actions in the plan. Limited and bounded resources (e.g. time, money, etc.) are debated as another cause of conflict between goals [Thangarajah et al., 2002]. Regardless of the cause of conflict, Riemsdijk et al. [2009] rightly pinpoint that to prevent the agent from pursuing conflicting goals, goals and their mutual conflicts must be *represented* in the first place. They broadly distinguish three approaches for representing conflicting goals: (i) providing the agent with an explicit representation of conflicting goals (e.g. [Pokahr et al., 2005]); (ii) equipping the agent with the capability to reason about the plans to satisfy the goals and consequently *inferring* the conflict between goals (e.g. [Thangarajah et al., 2003]); and (iii) representing goals using logic and considering them conflicting if they are logically speaking *inconsistent* (e.g. [Boer et al., 2002, 2007; Broersen et al., 2002; Riemsdijk et al., 2002; Toniolo, 2013]). Goals in this research are represented, using logic, as conjunctions of literals. We, therefore, use the widely acceptable approach in (iii) to represent conflicting goals as follows:

Definition 15 (Conflicting Goals)**.** *Goal g_i and g_j are in conflict iff satisfying one requires bringing about a state of affairs that is in conflict with the state of affairs required for satisfying the other. The set of conflicting goals is defined as:*

$$cf_{goal} = \{(g_i, g_j) \textit{ s.t. } \exists r \in g_i, \neg r \in g_j \textit{ or } \exists \neg r \in g_i, r \in g_j\} \tag{3.15}$$

Example 3.6. *Continuing Example 3.3, in which $strike$ was one of the agent's goals, we define another goal for the agent that is called $submission$. This goal requires the agent to go to office and finalise a project report to be submitted. So we have:*

$$strike = \{union\_member, \neg office, \neg meeting\_attended\}$$

$$submission = \{office, report\_finalised\}$$

Goals $strike$ and $submission$ are clearly in conflict since the former requires the agent not to go to office, while being present in the office in one of the requirements of the latter.

Goal-norm Conflict

Similar to the conflict between goals, conflict between goals and norms can either (i) be explicitly represented, for example in terms of integrity constraints, in which case there is not much computational formalism involved; or (ii) be inferred from plans that aim at satisfying goals and complying with norms. In this case, if there is no plan that satisfies goal g while not violating norm n, the agent establishes that there is a conflict between g and n; or (iii) be represented using logic. Despite much discussion in the literature, this type of conflict has rarely been computationally formulated. For example, López et al. [2005] talk about conflict between goals and norms in terms of goals being hindered by norms or vice versa, however, it is not clear what it means for a norm to hinder a goal (e.g. in what ways does compliance prevent goal achievement?). The same applies to the approach offered by Modgil and Luck [2008], who suggest a mechanism to resolve the conflicts between desires and normative goals. In this approach, norms are represented as system goals that may conflict with an agent's goals or desires. Social goals and individual goals do not need to conflict directly. Instead, conflict arises from the reward or punishment from complying with or violating a norm that may facilitate or hinder some of the agent's individual goals. Oren [2013] approach to normative practical reasoning, although it considers the conflict between goals and norms, does not formulate conflict, neither conceptually nor computationally. Conflict is instead *inferred* from paths or plans. Goal-norm incompatibility indeed only arises due to the fact that certain actions may satisfy one but not the other. Similar to the approach in Toniolo et al. [2011], we use logic to represent and formulate the conflict between goals and norms, but we differ in that we do not expect the agent to resolve the conflict by sacrificing its goals in favour of norms. Instead, similar to Oren [2013], preferences are used to determine whether the agent should comply with the norm or satisfy its goal.

Definition 16 (Conflicting Goal and Obligation)**.** *An obligation norm $n = \langle o, a_{con}, a_{sub}, dl \rangle$ and a goal g are in conflict, if executing action a_{sub} that is the subject of the obligation, brings about postconditions that are in conflict with the requirements of goal g. The set of conflicting goals and obligations is formulated as:*

$$cf_{goalobl} = \{(g, n), (n, g) \text{ s.t. } \exists r \in g, \neg r \in ps(a_{sub})^{-} \text{ or } \exists \neg r \in g, r \in ps(a_{sub})^{+}\} \tag{3.16}$$

Example 3.7. *Recall goal $strike = \{union\_member, \neg office, \neg meeting\_attended\}$ from Example 3.3 and obligation $n = \langle o, comp\_funding, attend\_meeting, 2 \rangle$ from Example 3.4. The postconditions of action $attend\_meeting$, that is the subject of obligation, are as follows: $ps(attend\_meeting) = \{meeting\_attended, summary\_documented\}$. Complying with the obligation brings about $meeting\_attended$ that prevents fulfilling $\neg meeting\_attended$ as one the requirements of goal $strike$. Goal $strike$ and norm n are therefore in conflict.*

Definition 17 (Conflicting Goal and Prohibition)**.** *A prohibition norm $n = \langle f, a_{con}, a_{sub}, dl \rangle$ and a goal g are in conflict, if the postconditions of a_{sub} contribute to satisfying g, but executing action a_{sub} is prohibited by norm n. The set of conflicting goals and prohibitions is formulated as:*

$$cf_{goalpro} = \{(g, n), (n, g) \text{ s.t. } \exists r \in g, r \in ps(a_{sub})^{+} \text{ or } \exists \neg r \in g, \neg r \in ps(a_{sub})^{-}\} \tag{3.17}$$

Example 3.8. *Recall goal $submission = \{office, report\_finalised\}$ from Example 3.6 and prohibition $n = \langle f, giving\_birth, work, 2 \rangle$ from Example 3.5. The postconditions of action*

$work$, that is the subject of prohibition, are as follows: $ps(work) = \{office, meeting\_attended\}$. Since this action is prohibited, $office$ cannot be brought about. However, $office$ is one of the requirements of goal $submission$, and hence the conflict between goal $submission$ and prohibition n.

Definition 18 (Conflicting Goal and Norm)**.** *The entire set of conflicting goals and norms is the union of conflicting goals and obligations, $cf_{goalobl}$, and conflicting goals and prohibitions, $cf_{goalpro}$, which gives $cf_{goalnorm}$:*

$$cf_{goalnorm} = cf_{goalobl} \cup cf_{goalpro} \tag{3.18}$$

Norm-norm Conflict

Similar to other types of conflict defined in this work, conflict between norms is also formulated using logic. Before proposing our formulation in Definitions 19, 20, and 21, we briefly survey the approaches that discuss normative conflicts in a practical reasoning context.

Oren et al. [2008] introduce a set of mutually exclusive norms that may not be complied with simultaneously. Norms do not have an internal structure in [Oren et al., 2008] and the conflict between them is detected externally and explicitly presented to the agent. Vasconcelos et al. [2009] offer a method for conflict detection and resolution between norms. A conflict is detected when an action is simultaneously prohibited and permitted/obliged and its variables have overlapping values, where variables specify the scope of influence of the norm. A detected conflict is resolved by manipulating the constraints associated with the norm variables to remove any overlap between their values. The aim of conflict resolution is to enable the agent to comply with the overall set of norms imposed on it. In [Criado et al., 2015], authors appreciate the importance of detecting the normative conflict dynamically. They propose a mechanism based on *coherence theory* [Criado et al., 2015], in which an agent dynamically computes a preference order over subsets of its competing norms by considering their coherence and inconsistencies. Also, in contrast to Vasconcelos et al. [2009], conflict in Thagard [2002] is not limited to conflict between a prohibition and an obligation or a permission. Eight variations of conflict are explored in total that cover all possible interactions of norms of type obligation, prohibition and permission. Giannikis and Daskalopulu [2011] are concerned with normative conflicts that arises for agents engaging in electronic contracts. They identify a set of six primitive patterns of normative conflicts, four of which arise as a result of the deontic qualification employed in the respective norms. The other two conflict patterns are caused by the relation between the actions that are qualified deontically in the respective norms (e.g. obligations to execute $actionX$ and $\neg actionX$). Existing approaches, with the exception of

[Criado et al., 2015], treat conflict between norms statically by using a predefined preference order that determines which norm should be followed in case two norms are inconsistent. However, in dynamic environments, it can be quite challenging to specify all inconsistencies that may occur. We therefore define the conflict between norms as dynamic, such that it depends upon the context of the plan in which the norms are activated. Given that norms modelled in this research are action-based and that we only model obligation and prohibition norms, we address two types of conflict: (i) two obligations are in conflict in plan π, iff they oblige the agent to execute two conflicting actions (see page 58) in an overlapping interval; and (ii) an obligation and prohibition are in conflict in plan π iff they oblige and prohibit the agent to and from executing the same action in an overlapping interval.

Definition 19 (Conflicting Obligations)**.** *Two obligation norms $n_i = \langle o, a_{con}, a_{sub}, dl\rangle$ and $n_j = \langle o, b_{con}, b_{sub}, dl'\rangle$ are in conflict in the context of sequence of actions π iff:*

(i) their activation conditions hold: $(a_{con}, t_{a_{con}}), (b_{con}, t_{b_{con}}) \in \pi$;

(ii) the obliged actions in n_i, i.e. a_{sub}, and n_2, i.e. b_{sub} have a concurrency conflict: $(a_{sub}, b_{sub}) \in cf_{action}$;

(iii) action a_{sub} is executed between the period that the activation condition of norm n_i holds and the deadline expires: $t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), t_{a_{con}} + d(a_{con}) + dl)$; and

(iv) action a_{sub} is in progress during the entire period over which the agent is obliged to execute action b_{sub}: $[t_{b_{con}} + d(b_{con}), t_{b_{con}} + d(b_{con}) + dl') \subseteq [t_{a_{sub}}, t_{a_{sub}} + d(a_{sub}))$

Figure 3-2 offers a graphical representation of this type of conflict. The set of conflicting obligations is denoted as cf^{π}_{oblobl} and formulated as:

$$\begin{aligned} cf^{\pi}_{oblobl} = \{(n_i, n_j) \text{ s.t. } & (a_{con}, t_{a_{con}}), (b_{con}, t_{b_{con}}) \in \pi; (a_{sub}, a_{sub}) \in cf_{action}; \\ & t_{a_{sub}} \in [t_{a_{con}} + d(a_{con}), t_{a_{con}} + d(a_{con}) + dl); \\ & [t_{b_{con}} + d(b_{con}), t_{b_{con}} + d(b_{con}) + dl') \subseteq [t_{a_{sub}}, t_{a_{sub}} + d(a_{sub}))\} \end{aligned} \tag{3.19}$$

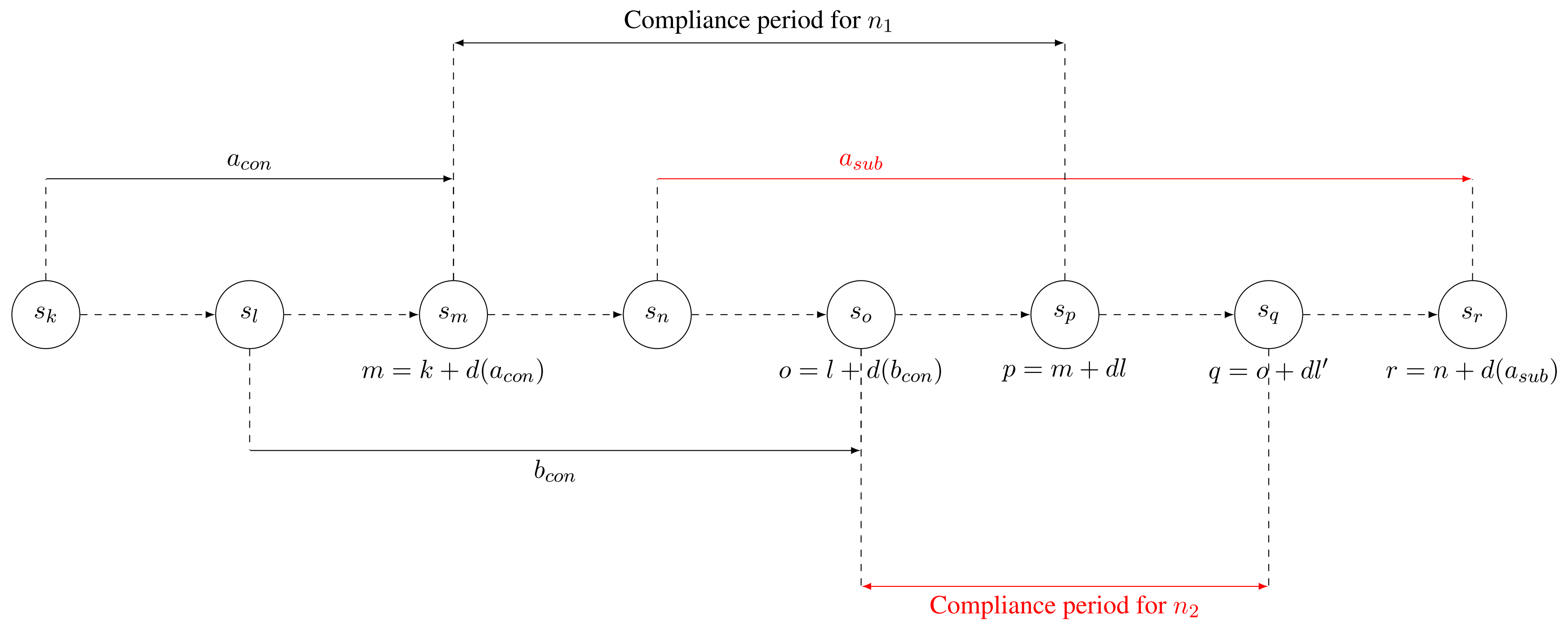


Figure 3-2: Conflict between Two Obligation Norms

Definition 20 (Conflicting Obligation and Prohibition)**.** *An obligation $n_i = \langle o, a_{con}, a_{sub}, dl\rangle$ and a prohibition $n_j = \langle f, b_{con}, a_{sub}, dl'\rangle$ are in conflict in the context of sequence of actions π iff:*

(i) their activation conditions hold: $(a_{con}, t_{a_{con}}), (b_{con}, t_{b_{con}}) \in \pi$; and

(ii) prohibition n_2 forbids the agent to execute action a_{sub} during the entire period over which obligation n_1 obliges the agent to take a_{sub}: $[t_{a_{con}} + d(a_{con}), t_{a_{con}} + d(a_{con}) + dl) \subseteq [t_{b_{con}} + d(b_{con}), t_{b_{con}} + d(b_{con}) + dl')$

Figure 3-3 displays a graphical representation of this type of conflict. The set cf^{π}_{oblpro} denotes the set of conflicting obligations and prohibitions as below:

$$cf^{\pi}_{oblpro} = \{(n_i, n_j), (n_j, n_i) \text{ s.t. } (a_{con}, t_{a_{con}}), (b_{con}, t_{b_{con}}) \in \pi; \\ [t_{a_{con}} + d(a_{con}), t_{a_{con}} + d(a_{con}) + dl) \subseteq [t_{b_{con}} + d(b_{con}), t_{b_{con}} + d(b_{con}) + dl')\} \quad (3.20)$$

Definition 21 (Conflicting Norms)**.** *All together, two sets cf^{π}_{oblobl} and cf^{π}_{oblpro} constitute the set of conflicting norms:*

$$cf^{\pi}_{norm} = cf^{\pi}_{oblobl} \cup cf^{\pi}_{oblpro} \quad (3.21)$$

As noted earlier, the conflict between norms is only detectable in the context of a sequence of actions. Thus, the examples for this type of conflict are provided in Chapter 5, where there is a comprehensive example that illustrates all different types of conflict.

3.2.3 Plans

Having defined sequences of actions and the properties and conflicts they can experience, we can now define which sequences of action can be identified as plans. In classical STRIPS-style planning a sequence of actions $\pi = \langle (a_0, 0), \cdots, (a_n, t_n)\rangle$ is a plan for $P = (\Delta, g, A)$, if all the fluents in Δ hold at time 0 and for each i, the preconditions of action a_i hold at time t_{a_i}, and goal g is satisfied in time m, where $m = Makespan(\pi)$. However, extending the conventional planning problem by multiple potentially conflicting goals and norms requires defining extra conditions in order to make a sequence of actions a plan and a solution for P. In what follows, we define what is required to identify a sequence of actions as a plan.

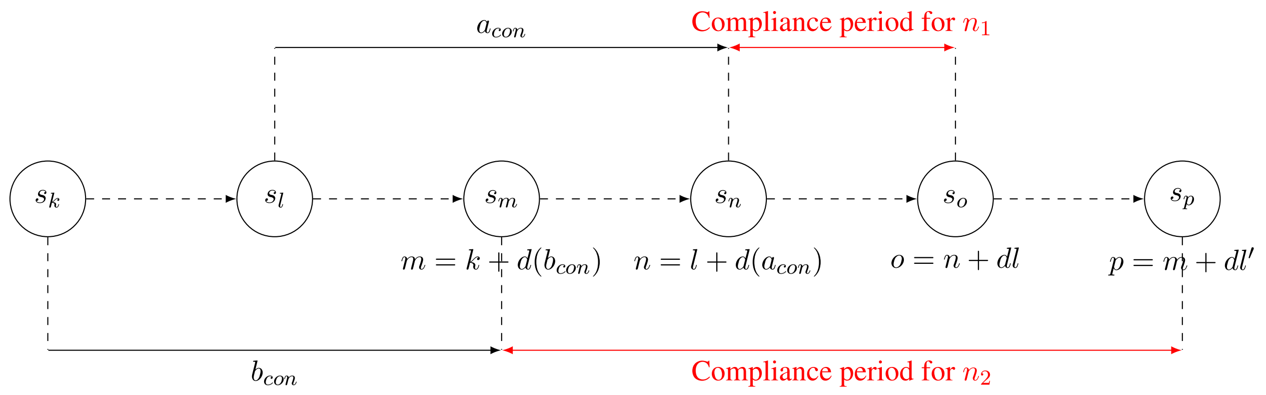


Figure 3-3: Conflict between an Obligation and a Prohibition

Definition 22 (Plan). *A sequence of actions $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ that brings about the sequence of states $S(\pi) = \langle s_0, \cdots s_m \rangle$, is a plan and solution for the normative planning problem $P = (FL, \Delta, A, G, N)$ iff the following six conditions hold:*

1. *all the fluents and only those fluents in Δ hold in the initial state:*

$$s_0 = \Delta \tag{3.22}$$

2. *the preconditions of action a_i holds at time t_{a_i} and throughout the execution of a_i:*

$$\forall k \in [t_{a_i}, t_{a_i} + d(a_i)), s_k \models pr(a_i) \tag{3.23}$$

3. *the set of goals satisfied by plan π is a non-empty consistent subset of goals:*

$$G_\pi \subseteq G \textit{ and } G_\pi \neq \emptyset \textit{ and } \nexists g_i, g_j \in G_\pi \textit{ s.t. } (g_i, g_j) \in cf_{goal} \tag{3.24}$$

4. *there is no concurrency conflict between actions that are executed concurrently:*

$$\nexists (a_i, t_{a_i}), (a_j, t_{a_j}) \in \pi \textit{ s.t. } t_{a_i} \leq t_{a_j} < t_{a_i} + d(a_i), (a_i, a_j) \in cf_{action} \tag{3.25}$$

5. *there is no conflict between norms complied with.*

$$\nexists n_i, n_j \in N_{cmp(\pi)} \textit{ s.t. } (n_i, n_j) \in cf^{\pi}_{norm} \tag{3.26}$$

6. *there is no conflict between goals satisfied and norms complied with:*

$$\nexists g \in G_\pi \textit{ and } n \in N_{cmp(\pi)} \textit{ s.t. } (g, n) \in cf_{goalnorm} \tag{3.27}$$

The set of plans for planning problem P such that they each meet all the six conditions above are denoted using Π.

3.3 Summary

In this chapter we formulated a formal model for normative practical reasoning that enables the agent to plan for multiple conflicting goals and norms simultaneously. Actions are durative and executable concurrently subject to the absence of concurrency conflict between them. In addition to concurrency conflict, three more types of conflict are explored in Section 3.2.2:

(i) conflict between goals; (ii) conflict between norms; and (iii) conflict between goals and norms. Finally in Section 3.2.3, we set out six conditions that identify a sequence of actions as a plan for the defined planning problem. The formal model defines all available plans for the agent. In the next chapter, we provide a computational implementation of the model that generates such plans.

Chapter 4

Identifying Plans via Answer Set Programming

Answer Set Programming (ASP) [Gelfond and Lifschitz, 1991] is a declarative programming paradigm, most commonly using logic programs under answer set semantics, which previously was referred to as stable semantics [Gelfond and Lifschitz, 1988]. In this paradigm, the user provides the description of a problem and ASP works out how to solve it by returning answer sets that correspond to problem solutions. A variety of programming languages for ASP exists, the most commonly used one of which is called *AnsProlog* (Programming in Logic with Answer sets) [Baral, 2003]. The existence of efficient algorithms, called solvers, to generate the answer sets to the provided problems has increased the number of applications[1] of ASP in different domains of autonomous agents and multi-agent systems such as planning (e.g. [Aker et al., 2013; Eiter et al., 2011; Lifschitz, 2002]), normative reasoning (e.g. [Balke et al., 2011; Cliffe, 2007; Li, 2014; Panagiotidi et al., 2012a]), model checking [Tang and Ternovska, 2005, 2007], agent reasoning [Blount and Gelfond, 2012; Gelfond, 2004]. The most widely used solvers at the moment are Clingo [Gebser et al., 2011] and DLV [Eiter et al., 1999]. In this work, we discuss the use of ASP to model and reason about the normative practical reasoning agents modelled in the previous chapter. First, we justify ASP as a suitable choice for this purpose. We then give an overview of AnsProlog syntax and semantics in Section 4.1, followed by the implementation of the model in Section 4.2.

In general, ASP and other non-monotonic logic programming systems such as Prolog [Colmerauer and Roussel, 1993] that use negation as failure ($not\ p$) to model negation, make an assumption that is referred to as the "closed world assumption" [Reiter and Kleer, 1987].

[1]We only mention the applications of ASP that are relevant to this thesis. The applications mentioned are therefore by no means comprehensive.

Based on this assumption *not* p is true if p cannot be proven true in the current program. This assumption makes it possible to model incomplete knowledge and reason about uncertainty which is an unavoidable part of modelling and reasoning about real world problems. While there are frameworks that formulate the agent reasoning problem using logic programming (e.g. [Alrawagfeh and Meneguzzi, 2014; Artikis et al., 2009]), there are a number of reasons why one should use ASP instead of other non-monotonic logic programming system such as Prolog. ASP is fully declarative and arguably more intuitive in contrast to the procedural nature of Prolog. The query-based nature of Prolog, that focuses on one issue at a time, makes it cumbersome to reason about different compositions of features that might hold or not in a logic program. Particularly, the practical reasoning problem in question makes ASP a better choice than Prolog, since it requires reasoning about all plans and their different qualities such as the goals they satisfy and the norms they obey or disobey. The semantics of ASP naturally provides all the alternative views of the world that are consistent with the logic program specified. In addition, there is evidence of ASP replacing Prolog implementations of formalisms for reasoning about actions, as a result of the existence of powerful solvers for ASP. The situation calculus [Reiter, 2001] is one of the most common formalisms for reasoning about actions and Prolog was first used to implement the situation calculus. However Lee and Palla [2010] later proposed a formulation of situation calculus in terms of the first-order stable model semantics which was then retransformed into ASP. More recently, the same authors, Lee and Palla [2014], proposed the formulation of the event calculus [Kowalski and Sergot, 1986] in the general theory of stable models which they then translated into ASP. Thus, as with the situation calculus, ASP solvers are used to compute the event calculus. The experiment conducted in [Lee and Palla, 2014] indicated that the ASP-based event calculus reasoner is significantly faster than other existing SAT-based [Gu et al., 1997] implementations [Mueller, 2004; Shanahan and Witkowski, 2004]. Apart from the situation and event calculus, action language $\mathcal{A}$ [Gelfond and Lifschitz, 1998] and its descendants (e.g. $\mathcal{B}, \mathcal{C}$ [Gelfond and Lifschitz, 1998]) are based on ASP. Temporal Action Logics (TAL) [Doherty et al., 1998] is another language for reasoning about actions that is implemented in ASP by Lee and Palla [2012]. All these implementations provide the indication that ASP is an appropriate tool for reasoning about actions. We therefore, in this chapter, propose an implementation of STRIPS [Fikes and Nilsson, 1971] as an action language in ASP. We then extend the implementation of STRIPS actions to implement the practical reasoning model described in the previous chapter. Given the above discussion, encoding a practical reasoning problem as a declarative logic program makes it possible to reason computationally about agent actions, goals and norms. This enables the agent to keep track of actions taken, goals satisfied and norms complied with or violated at each state of its evolution. More importantly, it provides the possibility of querying traces that fulfil certain requirements such as satisfying some specific goals. Consequently, instead of generating all possible traces and looking for those ones that satisfy a certain property (e.g. satisfy at least one goal), only those that *do* satisfy the property are generated.

4.1 AnsProlog Syntax and Semantics

As mentioned previously, a number of syntactic language representations for ASP exist. In this thesis we use AnsProlog which is one of the most common classes of these languages. and it has the following elements [Baral, 2003]:

Term: A term is a constant or a variable or a n-ary function $f(t_1, \cdots, t_n)$, where f is the function symbol and $t_1, \cdots, t_n$ are terms. Constants start with a lower-case letter, whereas variables start with an upper-case level. A term is *ground* if no variable occurs in it.

Atom: Atoms are the basic components of the language that can be assigned a truth value as true or false. An atom is a statement of form $A(t_1, \cdots, t_n)$, where A is a predicate symbol and $t_1, \cdots, t_n$ are terms.

Literal: Literals are atoms or negated atoms. Atoms are negated using *negation as failure* (not). $not\ a$ is true if there is no evidence proving the truth of a. An atom preceded by not is referred to as a naf-literal.

Herbrand universe of language $\mathcal{L}$ denoted as $HU_{\mathcal{L}}$ is the set of all ground terms which can be formed with the functions and constants in $\mathcal{L}$. The set of all ground atoms which can be formed with the functions, constants and predicates in $\mathcal{L}$ is called Herbrand base of language $\mathcal{L}$ and is denoted using $HB_{\mathcal{L}}$.

An AnsProlog program (e.g. Π) consist of a finite set of rules formed from atoms. The general rule syntax in AnsProlog is: $h_0 :- l_1, \cdots, l_m, not\ l_{m+1}, \cdots, not\ l_n.$, in which h_0 and l_is are atoms. h_0 is the rule head and $l_1, \cdots, l_m, not\ l_{m+1}, \cdots, not\ l_n$ are the body of the rule. The above rule is read as: h_0 is known/true, if $l_1, \cdots, l_m$ are known/true and none of l_{m+1}, l_n are known. If a rule body is empty, that rule is called a fact and if the head is empty it is called a constraint, indicating that the body of the the rule should not be satisfied. Another type of rules are called choice rules and are denoted as $l\{h_0, \cdots, h_k\}u :- l_1, \cdots, l_m, not\ l_{m+1}, \cdots, not\ l_n.$, in which h_is and l_is are atoms. l and u are integers and the default values for them is 0 and 1, respectively. A choice rule is satisfied if the number of atoms belonging to $\{h_0, \cdots, h_k\}$ that are true/known is between the lower bound l and upper bound u. In order to interpret a rule that contains variables, the rule has to be *grounded*. The grounding of each

```
human(alice).
human(adam).
mortal(X) :- human(X).
```

Figure 4-1: Program Π

```
human(alice).
human(adam).
mortal(alice) :- human(alice).
mortal(adam) :- human(adam).
```

Figure 4-2: Ground Version of Program Π

rule r in Π is then the set of all rules obtained from substitutions of elements of HU_{Π} for the variables in the rule r. By grounding all $r \in \Pi$, we obtain $ground(\Pi)$.

Example 4.1. *Take the example in Figure 4-1. $alice$ is a human, so is $adam$ and all humans are mortal. The Herbrand Universe of this program consists of the terms $alice$ and $adam$. Replacing the variable X in the third rule we obtain two more atoms: $mortal(alice)$ and $mortal(adam)$. The grounded version of the program presented in Figure 4-1 is displayed in Figure 4-2.*

The semantics of AnsProlog is defined in terms of *answer sets*. The answer sets of program Π, are defined in terms of the answer sets of the ground program $ground(\Pi)$. An AnsProlog program without any naf-literal is denoted as Ansprolog^{-not}. In other words, program Π is an Ansprolog^{-not} if for all rules in the program $m = n$. An answer set of an AnsProlog^{-not} program Π is a minimal subset (with respect to subset ordering) S of HB that is closed under $ground(\Pi)$. The approach to define the answer sets of an AnsProlog program Π is to take a candidate answer set S of the program and transform Π with respect to S to obtain an Ansprolog^{-not} denoted by Π^S. S is an answer set of Π if S is the answer set of AnsProlog^{-not} program Π^S. This transformation is referred to as Gelfond-Lifschitz [Gelfond and Lifschitz, 1988] transformation.

Given an AnsProlog program Π and a set S of atoms from HB_{Π}, the Gelfond-Lifschitz [Gelfond and Lifschitz, 1988] transformation Π^S is obtained by deleting:

1. each rule that has a $not\ L$ in its body with $L \in S$, and

2. literals of form $not\ L$ in the bodies of the remaining rules.

$$\boxed{\begin{array}{l} guilty :- evidence. \\ evidence :- trusted\_witness. \\ trusted\_witness :- not\ lying, witness. \\ witness. \\ believe :- not\ disbelieve. \\ disbelieve :- not\ believe. \\ lying :- disbelieve. \end{array}}$$

Figure 4-3: Program for Jury Example

The transformation (reduct) of choice rules was not a part of original Gelfond-Lifschitz transformation and was introduced later in [Lee et al., 2008]. Recently a simplified reduct for programs including choice rules is proposed by [Mark Law and Broda, 2015] as follows. Given an AnsProlog program Π - with choice rules- and a set S of atoms from HB_{Π}, the transformation Π^S is constructed in the following 4 steps:

1. Delete each rule that has a $not\ L$ in its body with $L \in S$.
2. Delete literals of form $not\ L$ in the bodies of the remaining rules.
3. for any choice rule r, $l\{h_0, \cdots, h_k\}u :- body(r)$, such that $l \leq |S \cap \{h_0, \cdots, h_k\}| \leq u$, replace r with the set of rules $\{h_i :- body^+(r) | h_i \in S \cap \{h_0, \cdots, h_k\}\}$.
4. for any remaining choice rules r, $l\{h_0, \cdots, h_k\}u :- body(r)$, replace r with the constraint $:- body^+(r)$.

After these transformation, the AnsProlog program Π is a program without any naf-literals and choice rules and it is therefore, an AnsProlog^{-not}, for which the answers are already defined. Informally, an answer set is a set of grounded atoms that satisfy the rules specifying the problem in a minimal and consistent fashion. Each answer set of the program corresponds to a solution for the problem encoded. If a program does not have any answer set, it said to be not satisfiable.

Example 4.2. *Consider the following example taken from [Cliffe, 2007] in which a jury wants to decide if the accused is guilty. The accused is guilty if there is evidence to support it. Evidence is a trusted witness. A witness is trusted if s/he is not lying. If the jury does not disbelieve the witness, it believes it and the other way around. Finally, a disbelieved witness is assumed to be lying. The situation is formulated in Figure 4-3.*

Using answer set semantics, the program in Figure 4-3 has two answer sets:
$Answer1 : \{guilty, evidence, trusted\_witness, witness, believe\}$

$Answer2 : \{witness, lying, disbelieve\}$
In $Answer1$, because the witness is not disbelieved, s/he is believed and because there is no evidence that is lying, s/he is trustworthy. Therefore, there exists evidence that the accused is guilty. In $Answer2$ however, the witness is disbelieved and therefore assumed to be lying, which does not qualify her/him as trustworthy and thus leaves no evidence for the accused to be guilty. In the next section we see how AnsProlog syntax and semantics are exploited for the purpose of modelling the practical reasoning problem discussed in the previous chapter.

4.2 Translating the Normative Practical Reasoning Model into ASP

In this section, we demonstrate how a planning problem $P = (FL, \Delta, A, G, N)$, defined in the previous chapter (page 56), can be mapped into an answer set program such that there is a one to one correspondence between solutions for the planning problem and the answer sets of the program. Each part of the translation is accompanied by examples. To distinguish between the original code and the code for examples, we do not number the lines for the latter.

4.2.1 States

In the previous chapter (page 63) we described the semantics of the planning problem $P = (FL, \Delta, A, G, N)$ over a set of states. The execution of a sequence of actions from a given starting state $s_0 = \Delta$ brings about a sequence of states $\langle s_0, \cdots s_m \rangle$ for every discrete time interval from 0 to m, where $m = Makespan(\pi)$. Assuming that there is a plan that uses all available actions such that none of them can be executed concurrently, the maximum number of states, q, results from sum of duration of all actions: $q = \sum_{i=1}^{n} d(a_i)$. The facts produced by Line 1 in Figure 4-4 provide the program with all available states. Atom `holdsat(`x, s`)` is used to express that fluent x holds in state s; Line 2 encodes the fluents that hold at initial state `holdsat(`$x, 0$`).` All the fluents that hold in a state, hold in the next state unless they are terminated (Line 3 to 4) using `terminated(X,S1).` In other words fluents are inertial (see Figure 4-5). `state(S1;S2)` in Line 4 is a standard replacement for the duplication of `state(S1)` and `state(S2).`

4.2.2 Actions

In the previous chapter (Page 57), we described an action a as a tuple composed of well-defined sets of literals $pr(a), ps(a)$ to represents a's preconditions and postconditions and a positive number $d(a) \in \mathbb{N}$ for its duration. Postconditions are further divided into a set of add postconditions, $ps(a)^+$, and a set of delete postconditions, $ps(a)^-$. The encoding for actions

$\forall\, k \in [0, q]$

```
1  state(k).
```

$\forall\, x \in \Delta$

```
2  holdsat(x,0).
```

Figure 4-4: Rules for State

```
3  holdsat(X,S2) :- holdsat(X,S1), not terminated(X,S1),
4                   state(S1;S2), S2=S1+1.
```

Figure 4-5: Inertial Fluents

is provided in Figure 4-6. Each durative action is encoded as `action(`a, d`)` (Line 5), where a is the name of the action and d is the duration. Recalling the previous chapter (page 57), the preconditions $pr(a)$ of action a hold in state s if $s \models pr(a)$. This is expressed in Line 6 using atom `pre(`a`,S)`, where $pr(a)^+$ and $pr(a)^-$ are positive and negative literals in $pr(a)$. In order to make the coding more readable we introduce the shorthand `EX(X,S)`, where `X` is a set of fluents that should hold at state `S`. For all $x \in$ `X`, `EX(X,S)` is translated into `holdsat(`x`,S)` and for all $\neg x \in$ `X`, `EX(`$\neg$`X,S)` is translated into `not EX(`x`,S)` using negation as failure. The agent has the choice to execute any of its actions in any state. This is expressed in choice rule presented in Line 7. As discussed in Section 4.1 (page 78), since there is no lower and upper bound expressed for `{executed(A,S)}`, the default value of `0{executed(A,S)}1` is implied, meaning that the agent has the choice whether to execute an action. Following the approach in [Blum and Furst, 1997] (see the previous chapter, page 57) we assume that the preconditions of a durative action should be preserved when it is in progress. We first encode the description of an action in progress, followed by ruling out the possibility of an action being in progress in the absence of its preconditions. A durative action is in progress, `inprog(A,S)`, from the state in which it begins to the state in which it ends (Line 8 to 9). Line 10, rules out the execution of an action, when the preconditions of the action do not hold during its execution. Another assumption made in Section 3.1.1, page 57, is that the agent cannot start two actions at the exact same time (Line 11 to 12). Once an action starts in one state, the result of its execution is reflected in the state where the action ends. This is expressed through (i) Line 13 that allows the add postconditions of the action to hold when the action ends, and (ii) Line 14 to 15 that allow the termination of the delete postconditions. The termination happens in the state before the end state of the action. The reason for this is the inertia of fluents that was expressed in Lines 3 and 4. Delete postconditions of an action

$\forall a \in A$ s.t. $d(a)$

```
action(a,d).
pre(a,S) :- EX(pr(a)+,S), not EX(pr(a)-,S), state(S).
```

```
{executed(A,S)} :- action(A,D), state(S).
inprog(A,S2) :- executed(A,S1), action(A,D), state(S1;S2),
                S1<=S2, S2<S1+D.
:- inprog(A,S), action(A,D), state(S), not pre(A,S).
:- executed(A1,S), executed(A2,S), A1!=A2,
   action(A1,D1), action(A2,D2), state(S).
```

$ps(a)^+ = X \Leftrightarrow \forall x \in X \cdot$

```
holdsat(x,S2) :- executed(a,S1), action(a,d), state(S1;S2), S2=S1+d.
```

$ps(a)^- = X \Leftrightarrow \forall x \in X \cdot$

```
terminated(x,S2) :- executed(a,S1), action(a,d), state(S1;S2),
                    S2=S1+d-1.
```

Figure 4-6: Rules for Translating Actions

```
action(attend_interview,4).
pre(attend_interview,S) :- holdsat(invitation,S), holdsat(venue,S),
                           state(S).
```

```
holdsat(interviewed,S2) :- executed(attend_interview,S1),
                           action(attend_interview,4), state(S1;S2),
                           S2=S1+4.
```

```
terminated(invitation,S2) :- executed(attend_interview,S1),
                             action(attend_interview,4), state(S1;S2),
                             S2=S1+4 − 1.
```

Figure 4-7: Implementation of Action $attend\_interview$

are terminated in the state before the end state of the action, so that they will not hold in the following state, in which the action ends (i.e. they are deleted from the state). Please note that the add postconditions can similarly be *initiated* in the state before the end state of the action. A fluent that is initiated in a state will then hold in the following state. However, currently, to reduce the grounding costs of the program, the add postconditions automatically appear at the end state without being initiated previously.

Example 4.3. *Figure 4-7 shows the implementation of action in Example 3.1, page 58:* $attend\_interview = \langle \{invitation, venue\}, \{interviewed, \neg invitation\}, 4\rangle$.

$\forall g \in G$

```
16  satisfied(g,S) :- EX(g^+,S), not EX(g^-,S), state(S).
```

Figure 4-8: Rules for Translating Goals

```
satisfied(strike,S) :- holdsat(union_member,S), not holdsat(office,S),
                       not holdsat(meeting_attended,S), state(S).
```

Figure 4-9: Implementation of Goal *strike*

4.2.3 Goals

From Chapter 3 (page 59), we have goal g is satisfied in state s if $s \models g$. This is expressed in Figure 4-8, Line 16, where g^+ and g^- are the positive and negative literals in set g. Positive literals should belong to the state, `EX(`g^+`,S)`, and the negative ones, `EX(`g^-`,S)`, should be absent so that atom `satisfied(`g`,S)` holds in state `S`.

Example 4.4. *Figure 4-9 shows the implementation of the goal in Example 3.3, page 60:* $strike = \{union\_member, \neg office, \neg meeting\_attended\}$.

4.2.4 Norms

The conditional action-based norms that are the focus of this research were discussed in the previous chapter, page 60. The encoding for norms is presented in Figure 4-10. Lines 17–31 deal with obligations and prohibitions of form[2]: $n = \langle o|f, a_{con}, a_{sub}, dl\rangle$. In order to implement the concepts of norm compliance and violation described in Chapter 3, page 63, we introduce a normative fluent, $o|f(n, a_{sub}, dl')$, that holds over the compliance period. Compliance period begins from the state in which action a_{con}'s execution ends. The compliance period then ends within dl time units of end of action a_{con}, which is denoted as dl' in the normative fluent. An obligation fluent $o(n1, a_{sub}, dl')$ denotes that action a_{sub}'s execution should begin before deadline dl' or be subject to violation, while prohibition fluent $f(n2, a_{sub}, dl')$ denotes that action a_{sub} should not begin before deadline dl' or be subject to violation. Lines 17–18 and 24–25 establish the obligation and prohibition fluents that hold for the duration of the compliance period.

In terms of compliance, if the obliged action *begins* during the compliance period in which the obligation fluent $o(n1, a_{sub}, dl')$ holds, the obligation is complied with (Line 19 to 20). The atom `cmp(o(`$n1$`,`a`,DL),S)` is used to indicate the compliance to norm $n1$ in state `S`[3].

[2]Since ASP syntax does not allow subscripts, a_{con} and a_{sub} appear as a_con and a_sub in the code.

[3]`DL` is a variable representing dl + `S2`.

$\forall n = \langle o|f, a_{con}, a_{sub}, dl \rangle \in N$

```
17 holdsat(o(n1,a_sub,dl+S2),S2) :- executed(a_con,S1), action(a_con,d1),
18                                                      S2=S1+d1,state(S1;S2).
19 cmp(o(n1,a,DL),S) :- holdsat(o(n1,a,DL),S), executed(a,S),
20                      action(a,d),state(S), S!=DL.
21 terminated(o(n1,a,DL),S) :- cmp(o(n1,a,DL),S), state(S).
22 vol(o(n1,a,DL),S) :- holdsat(o(n1,a,DL),S), DL=S, state(S).
23 terminated(o(n1,a,DL),S) :- vol(o(n1,a,DL),S), state(S).
24 holdsat(f(n2,a_sub,dl+S2),S2) :- executed(a_con,S1), action(a_con,d1),
25                                                     S2=S1+d1, state(S1;S2).
26 cmp(f(n2,a,DL),S) :- holdsat(f(n2,a,DL),S), action(a,d),
27                      DL=S, state(S).
28 terminated(f(n2,a,DL),S) :- cmp(f(n2,a,DL),S), state(S).
29 vol(f(n2,a,DL),S) :- holdsat(f(n2,a,DL),S), executed(a,S),
30                      state(S), S!=DL.
31 terminated(f(n2,a,DL),S) :- vol(f(n2,a,DL),S), state(S).
```

Figure 4-10: Rules for Translating Norms

The obligation fluent is terminated in the same state that compliance is detected (Lines 21). If the deadline expires and the obligation fluent still holds, it means that the compliance never occurred during the compliance period and norm n is therefore violated (Line 22). Atom `vol(o(n1,a,DL),S)` denotes the violation. The obligation fluent is terminated when the deadline expires and the norm is violated (Line 23). On the other hand, a prohibition norm is violated if the forbidden action *begins* during the compliance period in which the prohibition fluent $f(n2, a_{sub}, dl')$ holds (Line 29 to 30). As with the obligation norms, after being violated, the prohibition fluent is terminated (Line 31). If the deadline expires and the prohibition fluent still holds, that means the prohibited action did not begin during the compliance period and norm $n2$ is therefore complied with (Line 26 to 27). The obligation fluent is terminated in the same state that the compliance is detected (Line 28).

Example 4.5. *Figure 4-11 shows the implementation of the obligation in Example 3.4, page 62:* $n1 = \langle o, comp\_funding, attend\_meeting, 2 \rangle$.

Example 4.6. *Figure 4-12 shows the implementation of the prohibition in Example 3.5, page 62:* $n2 = \langle f, giving\_birth, work, 2 \rangle$.

4.3 Mapping of Answer Sets to Plans

Having implemented the components of $P = (FL, \Delta, A, G, N)$, in this section we encode the criteria for a sequence of actions to be identified as a plan and solution for P. These criteria

```
holdsat(o(n1,attend_meeting,2+S2),S2) :- executed(comp_funding,S1),
                     action(comp_funding,3), S2=S1+3, state(S1;S2).
cmp(o(n1,attend_meeting,DL),S) :- holdsat(o(n1,attend_meeting,DL),S),
                 executed(attend_meeting,S),action(attend_meeting,3),
                                    state(S), S!=DL.
terminated(o(n1,attend_meeting,DL),S) :- cmp(o(n1,attend_meeting,DL),S),
                                              state(S).
vol(o(n1,attend_meeting,DL),S) :- holdsat(o(n1,attend_meeting,DL),S),
                                         DL=S, state(S).
terminated(o(n1,attend_meeting,DL),S) :- vol(o(n1,attend_meeting,DL),S),
                                              state(S).
```

Figure 4-11: Implementation of Obligation $n1$

```
holdsat(f(n2,work,2+S2),S2) :- executed(giving_birth,S1), S2=S1+4,
                          action(giving_birth,4), state(S1;S2).
cmp(f(n2,work,DL),S) :- holdsat(f(n2,work,DL),S), action(work,1),
                                         DL=S, state(S).
terminated(f(n2,work,DL),S) :- cmp(f(n2,work,DL),S), state(S).
vol(f(n2,work,DL),S) :- holdsat(f(n2,work,DL),S), executed(work,S),
                                           state(S), S!=DL.
terminated(f(n2,work,DL),S) :- vol(f(n2,work,DL),S), state(S).
```

Figure 4-12: Implementation of Prohibition $n2$

(defined in the previous chapter (page 72)) are encoded in Figure 4-13. The rule in Line 33 is responsible for constraining the answer sets to those that fulfil at least one goal by excluding answers that do not satisfy any goal. The input for this rule is provided in Line 32, where goals are marked as satisfied if they are satisfied in at least one state. Line 34 prevents satisfying two conflicting goals, hence guaranteeing the consistency of satisfied goals in a plan (see Example 4.7). Preventing the concurrency of conflicting actions, is implemented in Line 35, by expressing that such two actions cannot be in progress together (see Example 4.8). Lines 36 and 37 provides the input for Lines 38, which excludes the possibility of satisfying a goal and complying with a norm that are conflicting (see Examples 4.9 and 4.10). Note that since norms are action-based, the implementation prevents complying with conflicting norms automatically: (i) if two obligations oblige the agent to execute two conflicting actions concurrently, one of them has to be violated since the concurrency of conflicting actions was already prevented in Line 35; and (ii) regarding conflicting obligation and prohibition, by definition, executing the obliged action and hence complying with the obligation causes the violation of the prohibition that prevents the execution of the very same action. Conversely, not executing the prohibited action and hence complying with the prohibition, results in the violation of the obligation.

```
satisfied(g) :- satisfied(g,S), state(S).
:- not satisfied(g1), ... , not satisfied(gm).
```

$\forall\ (g_1, g_2) \in cf_{goal}$

```
:- satisfied(g1),satisfied(g2).
```

$\forall\ (a_1, a_2) \in cf_{action}$

```
:- inprog(a1,S), inprog(a2,S), action(a1,d1), action(a2,d2), state(S).
```

```
complied(n1) :- cmp(o(n1,a,DL),S), state(S).
complied(n2) :- cmp(f(n2,a,DL),S), state(S).
```

$\forall\ (g, n) \in cf_{goalnorm}$

```
:- satisfied(g), complied(n).
```

Figure 4-13: Solutions for Problem P

```
satisfied(strike) :- satisfied(strike,S), state(S).
satisfied(submission) :- satisfied(submission,S), state(S).
```

```
:- satisfied(strike),satisfied(submission).
```

Figure 4-14: Implementation of Prevention of Conflicting Goals

Example 4.7. *Figure 4-14 shows the implementation of the conflicting goals from Example 3.6, page 67:* $strike = \{union\_member, \neg office, \neg meeting\_attended\}$ *and* $submission = \{office, report\_finalised\}$.

Example 4.8. *Figure 4-15 shows the implementation of the conflicting actions from Example 3.2, page 59:* $attend\_course = \langle\{fee\_paid\}, \{course\_attended, certificate\}, 3\rangle$ *and* $comp\_funding = \langle\{\neg fee\_paid, \neg money\}, \{fee\_paid\}, 4\rangle$.

Example 4.9. *Figure 4-16 shows the implementation of the conflicting goal and obligation of Example 3.7, page 68:* $strike = \{union\_member, \neg office, \neg meeting\_attended\}$ *and* $n = \langle o, comp\_funding, attend\_meeting, 2\rangle$. *The postconditions of action* $attend\_meeting$*, that is the subject of the obligation, are as follows:* $ps(attend\_meeting) = \{meeting\_attended, summary\_documented\}$. *Complying with the obligation brings about* $meeting\_attended$ *that prevents fulfilling* $\neg meeting\_attended$ *as one the requirements of goal* $strike$*.*

Example 4.10. *Figure 4-17 shows the implementation of conflicting goal:* $submission = \{office, report\_finalised\}$ *and prohibition:* $n = \langle f, giving\_birth, work, 2\rangle$ *of Example 3.8,*

```
:- inprog(attend_course,S), inprog(comp_funding,S), action(attend_course,5),
   action(comp_funding,3), state(S).
```

Figure 4-15: Implementation of Prevention of Conflicting Actions

```
satisfied(strike) :- satisfied(strike,S), state(S).
```

```
complied(n) :- cmp(o(n,attend_meeting,DL),S), state(S).
```

```
:- satisfied(strike), complied(n).
```

Figure 4-16: Implementation of Prevention of Conflicting Goals and Obligations

```
satisfied(submission) :- satisfied(submission,S), state(S).
```

```
complied(n) :- cmp(f(n,work,DL),S), state(S).
```

```
:- satisfied(submission), complied(n).
```

Figure 4-17: Implementation of Prevention of Conflicting Goals and Prohibitions

page 68. The postconditions of action $work$, that is the subject of prohibition, are as follows: $ps(work) = \{office, attend\_meeting\}$. Since this action is prohibited, $office$ cannot be brought about. However, $office$ is one of the requirements of goal $submission$, and hence the conflict between goal $submission$ and prohibition n.

Having encoded the criteria of a plan, we are now in a position to map the answers of the encoded program to the solutions of our planning problem. Let program Π_{base} consist of Lines 1–38. The following theorems state the correspondence between the solutions for problem P and answer sets of program Π_{base}.

Theorem 4.1. *Given a planning problem $P = (FL, I, A, G, N)$, for each answer set Ans of Π_{base} the set of atoms of the form `executed`(a_i, t_{a_i}) in Ans encodes a solution to the planning problem P.*

Proof. We first recall the definition of a plan from the previous chapter, Section 3.2.3 and then prove that program Π_{base} generates all sequences of actions that meet the criteria that identifies a sequence of actions as a plan. This implies that the sequence of actions that is a part of the answer set satisfies all the criteria to be a solution to the encoded planning program.

Actions and more precisely the postconditions of actions are what cause the change from one state to another one. Line 7 generates all sequences of actions. Line 13 changes a state in which some actions end by adding the add postconditions of those actions to the state. In contrast, Lines 14 and 15 terminate the delete postconditions of actions ending in the next state such that those postconditions do not hold in the following state. If there is no action ending in a state the state remains the same as the previous state, because all the fluents are inertial and they hold in the next state unless they are terminated (Line 4). A sequence of actions $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ is a plan and solution for normative planning problem $P = (FL, I, A, G, N)$ iff:

1. All the fluents in Δ hold in the initial state:

   $$s_0 = \Delta$$

   Line 2 ensures that all fluents in Δ are added to the initial state s_0.

2. The preconditions of action a_i holds at time t_{a_i} and throughout the execution of a_i:

   $$\forall k \in [t_{a_i}, t_{a_i} + d(a_i)), s_k \models pr(a_i)$$

   Lines 10 guarantees that the preconditions of an action hold all through its execution.

3. Set of goals satisfied by plan π is a non-empty consistent subset of goals:

   $$G_\pi \subseteq G \text{ and } G_\pi \neq \emptyset \text{ and } \nexists g_i, g_j \in G_\pi \text{ s.t. } (g_i, g_j) \in cf_{goal}$$

   Line 33 indicates that a non-empty subset of goals has to be satisfied in a plan, while Line 34 ensures the consistency of the goals satisfied.

4. There is no concurrency conflict between actions that are executed concurrently:

   $$\nexists (a_i, t_{a_i}), (a_j, t_{a_j}) \in \pi \text{ s.t. } t_{a_i} \leq t_{a_j} < t_{a_i} + d(a_i) \text{ and } (a_i, a_j) \in cf_{action}$$

   Preventing the concurrency conflict is encoded in Line 35.

5. There is no conflict between the norms complied with.

   $$\nexists n_i, n_j \in N_{cmp(\pi)} \text{ s.t. } (n_i, n_j) \in cf^{\pi}_{norm}$$

   As mentioned earlier, the implementation automatically prevents complying with conflicting norms.

6. There is no conflict between goals satisfied and norms complied with:

$$\nexists g \in G_\pi \text{ and } n \in N_{cmp(\pi)} \text{ s.t. } (g, n) \in cf_{goalnorm}$$

   Line 38 eliminates the possibility of conflict between goals satisfied and norms complied with.

Theorem 4.2. *Let* $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ *be a plan for* $P = (FL, I, A, G, N)$*, such that* $m = Makespan(\pi)$*. Then, there exists an answer set of* Π_{base} *containing atoms* $executed(a_i, t_{a_i})$ *with* $0 \leq t_{a_i} < m$ *that corresponds to* π*.*

Proof. Let the execution of sequence of actions in $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$ bring about the sequence of states $\langle s_0, \cdots s_q \rangle$. Let M_t be the set of following atoms (and nothing else):

$$\forall k, 0 \leq k \leq q \cdot M_t \models state(k) \tag{4.1}$$

$$\forall k, 0 \leq k \leq q \cdot x \in s_k \Rightarrow M_t \models holdsat(x, k) \tag{4.2}$$

$$\forall k, 0 \leq k < q \cdot x \in (s_k \setminus s_{k+1}) \Rightarrow M_t \models terminated(x, k) \tag{4.3}$$

$$\forall a \in A \cdot M_t \models action(a, d) \tag{4.4}$$

$$\forall k, 0 \leq k \leq q \cdot a \in A, pr(a)^+ \subseteq s_k, pr(a)^- \notin s_k \Rightarrow M_t \models pre(a, k) \tag{4.5}$$

$$\forall k, 0 \leq k < q \cdot (a, k) \in \pi \Rightarrow M_t \models executed(a, k) \tag{4.6}$$

$$\forall a. M_t \models executed(a, k),\ \forall k, t_a \leq k < t_a + d(a) \cdot M_t \models inprog(a, k) \tag{4.7}$$

$$\forall a. M_t \models executed(a, k),\ \forall x \in ps(a)^+ \cdot M_t \models holdsat(x, k + d(a)) \tag{4.8}$$

$$\forall a. M_t \models executed(a, k),\ \forall x \in ps(a)^- \cdot M_t \models terminated(x, k + d(a) - 1) \tag{4.9}$$

$$\forall k, 0 \leq k \leq q \cdot g \in G, g^+ \subseteq s_k, g^- \notin s_k \Rightarrow M_t \models satisfied(g, k) \tag{4.10}$$

$$\begin{aligned} &\forall n = \langle o, a_{con}, a_{sub}, dl \rangle \in N. M_t \models executed(a\_con, t_{a\_con}), \\ &\forall k, t_{a_{con}} + d(a_{con}) \leq k \leq t_{a_{con}} + d(a_{con}) + dl \cdot \\ &\qquad M_t \models holdsat(o(n, a\_sub, t_{a\_con} + d(a\_con) + dl), k) \end{aligned} \tag{4.11}$$

$$\exists k, 0 \leq k < q \cdot M_t \models holdsat(o(n,a,dl'),k), M_t \models executed(a,k), k! = dl' \Rightarrow$$
$$M_t \models cmp(o(n,a,dl'),k) \quad (4.12)$$

$$\exists k, 0 \leq k < q, M_t \models cmp(o(n,a,dl'),k) \Rightarrow M_t \models terminated(o(n,a,dl'),k) \quad (4.13)$$

$$\exists k, k = dl', M_t \models holdsat(o(n,a,dl'),k) \Rightarrow M_t \models vol(o(n,a,dl'),k) \quad (4.14)$$

$$\exists k, 0 \leq k \leq q, M_t \models vol(o(n,a,dl'),k) \Rightarrow M_t \models terminated(o(n,a,dl'),k) \quad (4.15)$$

$$\forall n = \langle f, a_{con}, a_{sub}, dl' \rangle \in N.M_t \models executed(a\_con, t_{a\_con}),$$
$$\forall k, t_{a_{con}} + d(a_{con}) \leq k \leq t_{a_{con}} + d(a_{con}) + dl' \cdot$$
$$M_t \models holdsat(f(n, a\_sub, t_{a\_con} + d(a\_con) + dl'), k) \quad (4.16)$$

$$\exists k, k = dl', M_t \models holdsat(f(n,a,dl'),k) \Rightarrow M_t \models cmp(f(n,a,dl'),k) \quad (4.17)$$

$$\exists k, 0 \leq k \leq q, M_t \models cmp(f(n,a,dl'),k) \Rightarrow M_t \models terminated(f(n,a,dl'),k) \quad (4.18)$$

$$\forall k, 0 \leq k < q \cdot M_t \models holdsat(f(n,a,dl'),k), M_t \models executed(a,k) \Rightarrow$$
$$M_t \models vol(f(n,a,dl'),k) \quad (4.19)$$

$$\exists k, 0 \leq k \leq q, M_t \models vol(f(n,a,dl'),k) \Rightarrow M_t \models terminated(f(n,a,dl'),k) \quad (4.20)$$

$$\exists k, 0 \leq k \leq q, M_t \models satisfied(g,k) \Rightarrow M_t \models satisfied(g) \quad (4.21)$$

$$\exists k, 0 \leq k \leq q, M_t \models cmp(o(n,a,dl'),k) \Rightarrow M_t \models complied(n) \quad (4.22)$$

$$\exists k, 0 \leq k \leq q, M_t \models cmp(f(n,a,dl'),k) \Rightarrow M_t \models complied(n) \quad (4.23)$$

We need to prove that M_t is an answer set of Π_{base}. Therefore, we need to demonstrate that M_t is a minimal model for $\Pi_{base}^{M_t}$. Let $r \in \Pi_{base}^{M_t}$ be an applicable rule. In order for M_t to be a model of $\Pi_{base}^{M_t}$, we need to show that r is applied (i.e. $M_t \models Head(r)$). We will go through each rule in the same order in Lines 1–38.

- r is of type rule in Line 1: fact and automatically applied.
- r is of type rule in Line 2: fact and automatically applied.

- r is of type rule in Lines 3–4: because of Gelfond-Lifschitz transformation, we know that $not\ terminated(x, s)$ is removed from this rule. combination of 4.2 and 4.3 for x at k gives $M_t \models holdsat(x, k + 1)$.

- r is of type rule in Line 5: fact and automatically applied.

- r is of type rule in Line 6: after Gelfond-Lifschitz transformation, from the body and description of this rule we have $a \in A$ and $pr(a)^+ \in s_k$ and $pr(a)^- \notin s_k$, which with 4.5 implies that $M_t \models pre(a, k)$.

- r is of type rule in Line 7: any action $a \in A$ can be executed in a state. After the transformation for choice rules, we obtain $\forall(a, k) \in \pi$ we have $executed(a, k) : -\ action(a, d)$, $state(k)$. and $\forall(a, k)$ s.t. $(a, k) \notin \pi$ we have $: -\ action(a, d), state(k)$. With 4.6, we know that $M_t \models executed(a, k)$.

- r is of type rule in Lines 8–9: $inprog$ atoms originate from execution of actions. From 4.6 we know that $\forall(a, k) \in \pi, M_t \models executed(a, k)$. Since a is executed with 4.7 we have $\forall k, t_a \leq k < t_a + d(a), M_t \models inprog(a, k)$.

- r is of type rule in Line 10: the head of this rule is empty. Since π is a plan and we have assumed the preconditions of actions in a plan are hold while the actions are in progress, this rule is applied.

- r is of type rule in Lines 11–12: the head of this rule is empty. Because π is a plan and we have assumed that two actions in a plan cannot have the exact same start state, this rule is applied.

- r is of type rule in Line 13: the body of this rule implies that the add postconditions of an executed action a hold in the state in which the action ends. Since $\forall(a, k) \in \pi, M_t \models executed(a, k)$, with 4.8 we have $\forall x \in ps(a)^+ \cdot M_t \models holdsat(x, k + d(a))$.

- r is of type rule in Lines 14–15: the body of this rule implies that the delete postconditions of an executed action a are terminated in the state before the end state of the action. Since $\forall(a, k) \in \pi, M_t \models executed(a, k)$, with 4.9 we have $\forall x \in ps(a)^- \cdot M_t \models terminated(x, k + d(a) - 1)$.

- r is of type rule in Line 16: after Gelfond-Lifschitz transformation, from the body and description of this rule we have $g^+ \in s_k$ and $g^- \notin s_k$, which with 4.10 implies that $M_t \models satisfied(g, k)$.

- r is of type rule in Lines 17–18: the body of the rule implies that normative fluents for obligations are hold over the compliance period if the action that is the condition of the norm is executed. If a_{con} for an obligation norm belongs to π, then based on 4.6 we know that $M_t \models executed(a\_con, t_{a\_con})$. From 4.6 and 4.11 we know that $M_t \models holdsat(o(n, a\_sub, t_{a\_con} + d(a\_con) + dl'), k)$ over the period $t_{a_{con}} + d(a_{con}) \leq k \leq t_{a_{con}} + d(a_{con}) + dl'$.

- r is of type rule in Lines 19–20: this rule expresses that if the obliged action is executed while the normative fluent holds, the norm is complied with. If a_{sub} is executed in π ($M_t \models executed(a\_con, t_{a\_con})$) while the normative fluent in 4.11 holds, with 4.12 we know that M_t models the compliance atom.

- r is of type rule in Line 21: complied obligations are terminated in the compliance state. With 4.12 we know that M_t models compliance atoms, and 4.13 implies that they are terminated in the same state.

- r is of type rule in Line 22: if the obligation fluent still holds when the deadline occurs, the obligation is violated. 4.14 implies this is modelled by M_t.

- r is of type rule in Line 23: violated obligations are terminated in the violation state. With 4.14 we know that M_t models violation atoms, and 4.15 implies that they are terminated.

- r is of type rule in Lines 24–25: the body of the rule implies that normative fluents for prohibition norms are hold over the compliance period if the action that is the condition of the norm is executed. If a_{con} for a prohibition belongs to π, then based on 4.6 we know that $M_t \models executed(a\_con, t_{a\_con})$. From 4.6 and 4.16 we know that $M_t \models holdsat(f(n, a\_sub, t_{a\_con} + d(a\_con) + dl'), k)$ over the period $t_{a_{con}} + d(a_{con}) \leq k \leq t_{a_{con}} + d(a_{con}) + dl'$.

- r is of type rule in Lines 26–27: this rule expresses that if the normative fluent still holds at the end of compliance period, the prohibition is complied with. 4.17 implies that M_t models the head of this rule.

- r is of type rule in Line 28: complied prohibitions are terminated in the compliance state. With 4.17 we know that M_t models compliance atoms, and 4.18 implies that they are terminated and this rule is applied.

- r is of type rule in Lines 29–30: this rule expresses that if the prohibited action is executed while the normative fluent holds, the norm is violated. If a_{sub} is executed in π ($M_t \models$

$executed(a\_sub, t_{a\_sub}))$ while the normative fluent in 4.16 holds, with 4.19 we know that M_t models the violation atom.

- r is of type rule in Line 31: violated prohibitions are terminated in violation state. With 4.19 we know that M_t models violation atoms, and 4.20 implies that they are terminated.
- r is of type rule in Line 32: this rule is applicable whenever a goal is satisfied in a state. With 4.10 and 4.21 we can obtain this ($M_t \models satisfied(g)$).
- r is of type rule in Line 33: the head of this rule is empty. Because π is a plan it has to satisfy at least one goal so, this rule is applied.
- r is of type rule in Lines 34: the head of this rule is empty. Because π is a plan and a plan cannot satisfy conflicting goals, this rule is applied.
- r is of type rule in Line 35: the head of this rule is empty. Because π is a plan and a plan cannot cannot contain concurrent execution of actions, this rule is applied.
- r is of type rule in Line 36: this rule is applicable whenever an obligation norm is satisfied in a state. With 4.12 and 4.22 we can obtain this ($M_t \models complied(n)$).
- r is of type rule in Line 37: similar reasoning as above, but withWith 4.17 and 4.3 for a prohibition norm.
- r is of type rule in Lines 38: the head of this rule is empty. Because π is a plan and a plan cannot satisfy conflicting goals and norms, this rule is applied.

By showing that every rule is applied, we have shown that M_t is a model for $\Pi_{base}^{M_t}$.

Now, we need to show that M_t is minimal, which means that there exist no other model of $\Pi_{base}^{M_t}$ that is a subset of M_t.

Let $M \subset M_t$ be a model for $\Pi_{base}^{M_t}$, then there must exist an atom $s \in (M_t \setminus M)$. If s is an atom that is generated because it is a fact, it must belong to M too and if that is not the case, then M cannot be a model. We now proceed with the rest of atoms that do not appear as facts:

- $s = executed(a, k)$: $M_t \models s$ implies that $r : executed(a, k) :- action(a, d), state(k).$, $r \in \Pi_{base}^{M_t}$. If $M \not\models executed(a, k)$, then r was applicable but not applied, therefore, M is not a model.
- $s = holdsat(x, k)$: $M_t \models s$ implies that one of the following four condition must have occurred:

  - 4.2: if this is the case then s was true in s_{k-1} (not terminated is removed from the body of this rule because of the Gelfond-Lifschitz). In this case the construction of M_t guarantees $M_t \models holdsat(x, k-1)$. If $M \not\models holdsat(a, k)$ then rule in Lines 3–4 are applicable but not applied, so M cannot be a model.
  - 4.8: if this is the case then we know that $M_t \models executed(a, k)$. Earlier on we showed that if $M_t \models executed(a, k)$, then $M \models executed(a, k)$ too. Thus, if $M \not\models holdsat(x, k + d(a))$ for all $x \in ps(a)^+$, then rule in Line 13 is applicable but not applied, so M cannot be a model.
  - 4.11: if this is the case then $x = o(n, a\_sub, t_{a\_con} + d(a\_con) + dl)$. Because M_t is a model then $M_t \models executed(a\_con, t_{a\_con})$. So $M \models executed(a\_con, t_{a\_con})$ too. Therefore, rule in Lines 17–18 is applicable and if $M \not\models s$ then this rule is not applied and M cannot be a model.
  - 4.16: if this is the case then $x = f(n, a\_sub, t_{a\_con} + d(a\_con) + dl)$. Similar to reasoning above but instead of rule in Lines 17–18, rule in Lines 24–25 is applicable.

- $s = pre(a, k)$: $M_t \models s$ implies that $\forall x \in pr(a)^+, x \in s_k$ and $\forall x \in pr(a)^-, x \notin s_k$. If M is a model then according to the transformed version of rule 6 $M \models pre(a, k)$ and if that is not the case then M is not a model.

- $s = inprog(a, k)$: $M_t \models s$ implies that $M_t \models executed(a, t_a)$. Therefore, $M \models executed(a, t_a)$. That means the rule in Lines 8–9 is applicable and if $\forall t_a \leq k < t_a + d(a), M \not\models inprog(a, k)$ then this applicable rule is not applied and therefore M cannot be a model.

- $s = cmp(o(n, a, dl'), k)$: $M_t \models s$ implies that $M_t \models holdsat(o(n, a, dl'), k)$ and also $M_t \models executed(a, k)$. Consequently, $M \models holdsat(o(n, a, dl'), k)$ and $M \models executed(a, k)$. As a result, rule in Lines 19–20 is applicable and if $M \not\models s$, the rule is not applied and M is not a model.

- $s = vol(o(n, a, dl'), k)$: $M_t \models s$ implies that $M_t \models holdsat(o(n, a, dl'), k)$ and also $k = dl'$. Because M is a model we know that $M \models holdsat(o(n, a, dl'), k)$. As a result rule in Line 22 is applicable and if $M \not\models vol(o(n, a, dl'), k)$, the rule is not applied and M is not a model.

- $s = cmp(f(n, a, dl'), k)$: $M_t \models s$ implies that $M_t \models holdsat(f(n, a, dl'), k)$ and also $k = dl'$. Because M is a model we know that $M \models holdsat(f(n, a, dl'), k)$ too. As a result rule in Lines 26–27 is applicable and if $M \not\models \mathit{cmp}(f(n, a, dl'), k)$, the rule is not applied and M is not a model.

- $s = \mathit{complied}(n)$: $M_t \models s$ because $M_t \models \mathit{cmp}(o(n, a, dl'), k)$ or because $M_t \models \mathit{cmp}(f(n, a, dl'), k)$. If $M_t \models \mathit{cmp}(o(n, a, dl'), k)$, $M \models \mathit{cmp}(o(n, a, dl'), k)$ too, therefore rule 36 is applicable and $M \models \mathit{complied}(n)$ or it is not a model. If $M_t \models \mathit{cmp}(f(n, a, dl'), k)$, $M \models \mathit{cmp}(f(n, a, dl'), k)$ too, therefore rule in Line 37 is applicable and $M \models \mathit{complied}(n)$ or it is not a model.

- $s = \mathit{vol}(f(n, a, dl'), k)$: $M_t \models s$ implies that $M_t \models \mathit{holdsat}(f(n, a, dl'), k)$ and also $M_t \models \mathit{executed}(a, k)$. Consequently, we know that $M \models \mathit{holdsat}(f(n, a, dl'), k)$ and $M \models \mathit{executed}(a, k)$. As a result rule in Lines 29–30 is applicable and if $M \not\models \mathit{vol}(f(n, a, dl'), k)$, the rule is not applied and M is not a model.

- $s = \mathit{terminated}(x, k)$: $M_t \models s$ implies that one of the following five situations:
  - 4.9: if this is the case then $M_t \models \mathit{executed}(a, k - d(a) + 1)$. If M is a model $M \models \mathit{executed}(a, k - d(a) + 1)$. Thus, the rule in Lines 14–15 is applicable and if $M \not\models \mathit{terminated}(s, k)$ then the rule is applicable but not applied, which means that M is not a model.
  - 4.13 and 4.15: if this is the case $x = o(n, a, dl') \in s_k$, and $M_t \models \mathit{cmp}(o(n, a, dl'), k)$ or $M_t \models \mathit{vol}(o(n, a, dl'), k)$. Since M is a model, then if $M_t \models \mathit{cmp}(o(n, a, dl'), k)$ the same applies to M and if $M_t \models \mathit{vol}(o(n, a, dl'), k)$ again the same applies to M. In either cases according to rules 21 and 23 $M \models \mathit{terminated}(o(n, a, dl'), k)$ or M is not a model.
  - 4.18 and 4.20: if this is the case $x = f(n, a, dl') \in s_k$, and $M_t \models \mathit{cmp}(f(n, a, dl'), k)$ or $M_t \models \mathit{vol}(f(n, a, dl'), k)$. Since M is a model, if $M_t \models \mathit{cmp}(f(n, a, dl'), k)$ the same applies to M and if $M_t \models \mathit{vol}(f(n, a, dl'), k)$ again the same applies to M. In either cases according to rules 28 and31 $M \models \mathit{terminated}(f(n, a, dl'), k)$ or M is not a model.

- $s = \mathit{satisfied}(g, k)$: $M_t \models s$ implies that $\forall x \in g^+, x \in s_k$ and $\forall x \in g^-, x \notin s_k$. If M is a model then according to the transformed version of rule 16 $M \models \mathit{satisfied}(g, k)$ and if that is not the case then M is not a model.

- $s = \mathit{satisfied}(g)$: $M_t \models s$ because $M_t \models \mathit{satisfied}(g, k)$. Since $M_t \models \mathit{satisfied}(g, k)$, according to previous item $M \models \mathit{satisfied}(g, k)$ too, therefore rule 32 is applicable and $M \models \mathit{satisfied}(g)$ or it is not a model.

The combination of these items demonstrates that M cannot be a model for $\Pi_{base}^{M_t}$ if it differs from M_t. M_t is therefore a minimal model for $\Pi_{base}^{M_t}$ and an answer set for Π_{base}.

4.3.1 Optimal Plans

Planning is a search problem that tries to find a sequence of actions that satisfy certain properties. The bigger the state space of the search, the more difficult it becomes to find a solution for such a search problem. Therefore, over the time different techniques such as defining heuristic functions, and decomposing the search space are introduced to mitigate the computational cost of such a search and seek optimal plans. Optimal solutions for a planning problem can be defined differently for example with respect to the cost of the plan, timespan of the plan or the number of actions involved in the plan. They can also be defined to capture certain features of the problem modelled. For instance, some planning problems require repetition of the same actions (e.g. bomb clearing scenario), while in others the actions are unique and are not to be repeated. Thanks to ASP's built-in optimisation functions, the capability of ASP to plan with the possibility of guaranteed optimality with respect to some criteria is widely exploited [Aker et al., 2013; Eiter et al., 2011; Erdem et al., 2013; Lifschitz, 2002]. So far the implementation proposed in this chapter provides the agent with all possible sequences of actions that are identified as a plan. What we are trying to do in this section is to define some restrictions that filters out some plans and hence leaves fewer plans for the agent to reason about.

Similar to Erdem et al. [2013], we define optimised plans as plans in which the agent does not repeat any action and it is not idle at any point in time. Both these criteria are merely based on the problems we are interested in modelling. For the first criterion, we simply use a constraint that is encoded in Lines 39–40 in figure 4-18. For the latter criterion, our strategy is first to identify the final state in each plan and then express that, before reaching the final state, there has to be at least one action in progress in every single state. But how to recognise the final state of a plan? Classical planning problems deal with a single goal. The final state in a solution corresponding to a classical planning problem is the state in which the goal is satisfied. However, in the planning problem introduced in this research, the agent needs to plan for a set of potentially inconsistent goals. Therefore, each solution may satisfy more than one goal. The final state of a solution satisfying more than one goal, is the state that holds at the latest time in which a goal is satisfied. To identify the final state, we first mark the state in which a goal is satisfied for the first time by a flag (Lines 41) using aggregate `min`. This aggregate is an operation on a set of weighted literals that evaluates to some value that is the minimum with respect to the rest of weights in the set: $\#min[L_1 = w_1, \cdots, L_n = w_n]$. We assign `S` to the weight of literal `satisfied(`g`,S)`: `[satisfied(`g`,S)=S]`, and use

```
39 :- executed(A,S1), executed(A,S2), action(A,D),
40    S1!=S2, state(S1;S2).
```

$\forall g \in G$

```
41 flag(M) :- M = #min[satisfied(g,S)=S], satisfied(g).
```

```
42 final(F) :- F = #max[flag(S) = S].
43 alpha(S) :- inprog(A,S), action(A,L), state(S).
44 :- final(S2), not alpha(S1), state(S1;S2),  S1<S2.
45 :- final(S1), alpha(S2), state(S1;S2),  S2>=S1.
```

Figure 4-18: Optimisation rules

`#min[satisfied(`g`,S))=S]` to find the minimum weight for this literal, which in fact is the first state in which the goal is satisfied. Note that the extra atom `satisfied(`g`)` is to ensure that the state in which a goal is satisfied for the first time is sought after, only if the goal is satisfied at least once. Then, in Line 42 we identify the state when the last flag is observed by using aggregate `max`. This aggregate, similar to `min`, is an operation on a set of weighted literals, however, it evaluates to some value that is the maximum with respect to the rest of wights in the set: $\#max[L_1 = w_1, \cdots, L_n = w_n]$. We assign `S` to the weight of literal `flag(S)`: `[flag(S)=S]`, and use `#max[flag(S)=S]` to find the maximum weight for this literal, which in fact is the latest state in which a flag holds. This state is the final state `final(S)`. We do not want the agent to be idle at any point of time. We therefore need to exclude the possibility of the existence of states in which there is no action in progress. `alpha(S)` in line 43 marks those states in which an action is in progress. Line 44 makes use of `alpha(S)` to prevent the agent from being idle before reaching the final state. Finally, Line 45 ensures that no action is in progress after the final state is reached in a plan.

Now, assume that program $\Pi = \Pi_{base} \cup \Pi^*$, where Π^* consists of lines 39–45. The following theorem states the correspondence between the optimised solutions for problem P and answer sets of program Π.

Theorem 4.3. *Given a planning problem $P = (FL, I, A, G, N)$, for each answer set Ans of Π the set of atoms of the form $executed(a_i, t_{a_i})$ in S encodes an optimal solution to the planning problem P. Conversely, each solution to the problem P corresponds to a single answer set of Π.*

Proof. Follows immediately from the structure of program Π^*.

4.4 Summary

In this chapter, we set out an implementation of the model described in the previous chapter. The computational tool is ASP. We first explained why ASP is an appropriate tool for our purpose, and then explained the syntax and semantics of ASP language, AnsProlog. The implementation itself is composed of the encoding for the different components of the model, the criteria that identify a sequence of actions as a plan, and the constraints on plans to ensure optimal plans.

The proposal of actions and planning problem in the previous chapter is based on STRIPS [Fikes and Nilsson, 1971]. Thus, this chapter effectively proposes an implementation of STRIPS as an action language in ASP, including the extensions made to STRIPS, namely multiple goals, durative actions and norms. These extensions were made to capture the features of normative practical reasoning problem modelled. Also the formulation of extensions was done with respect to the computational tool, thus there are no conceptual gaps between the formal model and its implementation to bridge. The most important distinction of this translation to the other translation of actions languages in ASP such as [Lee and Palla, 2012, 2014] is the fact that following the formal model, the translation accommodates multiple goals, norms and reasoning about the norms. Although implementing agents that are capable of reasoning about norms was previously considered in ASP [Panagiotidi et al., 2012a,b], to the best of our knowledge, our implementation is the first one that takes the duration of actions into account when reasoning about norm compliance and violation.

Chapter 5

Identifying the Best Plan

In Chapters 3 and 4 we provided a formal model and its implementation for an agent that is capable of planning for multiple goals and norms. However, the conflict between these goals and norms often makes it impossible for the agent to satisfy all its goals in a plan while complying with all the norms triggered in that plan. The agent therefore needs to reason about all these conflicts and where available the preferences between the conflicting entities, in order to decide on the best plan to execute. Given the complications of decision-making in such an environment it is very difficult for humans to understand such frameworks and their outcomes. Ideally, what the agent requires is a mechanism that allows decision-making with respect to existing inconsistencies in its attitude, while making the decision is understandable and explainable to others.

Reasoning about inconsistency and decision-making have both been studied in AI at length, however they have often been treated separately. Argumentation as a discipline deals distinctly with issues of handling inconsistency [Amgoud and Vesic, 2010; Dung, 1995; Prakken and Sartor, 1997] and decision-making [Amgoud and Prade, 2004b; Bonet and Geffner, 1996]. However, more recently, it has been argued [Amgoud, 2012; Amgoud and Prade, 2009] that inference about consistency is a part of decision-making. This line of reasoning results in the development of argumentation-based approaches that capture the issues of inconsistency and decision-making in the same framework [Amgoud, 2012; Amgoud and Prade, 2009], such that the decisions made are *justified* with respect to the inconsistencies. In addition, the dialogical aspect of argumentation makes it possible to explain the justifiability of a decision.

In the same manner, we propose an argumentation framework based on argumentation schemes and critical questions that enables the agent to argue over plan proposals with respect to (i) the conflicts between and within goals and norms; and (ii) the preferences between these entities. Arguing over a plan involves putting forward the plan as a proposal and letting the

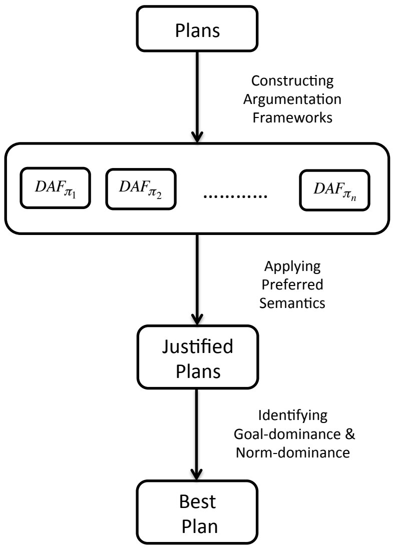

Plans
Constructing Argumentation Frameworks
DAF_{π_1}
DAF_{π_2}
....
DAF_{π_n}
Applying Preferred Semantics
Justified Plans
Identifying Goal-dominance & Norm-dominance
Best Plan

Figure 5-1: The Process of Identifying the Best Plan

agent to question the justifiability of the plan proposal by investigating why a certain goal is not satisfied in the proposed plan, or why a certain norm is violated. The evaluation of argumentation frameworks for the plan proposals results in identifying justified plans. The justified plans are further refined in a search for the best plan, by comparing the quality (i.e. preferences) and quantity (i.e. numbers) of goals satisfied and norms violated in these plans. In comparison of two plans, we refer to the plan that satisfies more important goals or more number of goals as *goal-dominant*. Conversely, a plan that violates more important norms or more number of norms is referred to as *norm-dominant*. Figure 5-1 displays a graphical representation of the process of identifying the best plan.

This chapter is organised as follows. In Section 5.1 we discuss the construction of an argumentation framework for each plan proposal. The evaluation of this framework toward identifying justified plans is discussed in Section 5.1.2, followed by investigating the properties of justified plans in Section 5.1.3. Section 5.2 focuses on identifying the best plan, followed by an illustrative example in Section 5.3. A discussion and summary of the chapter is provided in Sections 5.4 and 5.5.

5.1 Justified Plans

In this section, we show how to construct an argumentation framework that allows checking the justifiability of plans with respect to conflicts and preferences, as a step toward identifying the best plan(s). What arguments are justified in an argumentation framework is a subjective matter and can be defined differently depending on the argumentation semantics used for evaluating the arguments. We use the preferred semantics to determine the justifiability of plans. The rationale behind this choice is discussed in detail in Section 5.1.2, where plans are defined as justified only if they sacrifice the satisfaction of a goal or a norm for an equally or more important goal or norm. The plan is otherwise unjustified.

5.1.1 Argumentation Framework

In chapter 2 (page 35) we provided a survey of argumentation frameworks originating from Dung's argumentation framework [Dung, 1995]. With all their differences, they are similar in defining an argumentation framework as a set of arguments and a set of attacks between them [Dung, 1995]: $AF = \langle Arg, Att \rangle, Att \subseteq Arg \times Arg$. In this section we discuss the choice of an argumentation framework that is appropriate to model the arguments and attacks built to evaluate plan proposals. Another factor we want to consider in such an argumentation framework is the preferences between these arguments. Preferences are introduced and frequently used in non-monotonic logics to model human modes of reasoning where explicit expression of preferences is an inevitable part of the process. Since argumentation seeks to be another non-monotonic formalism, to have equivalent representational power, preferences must be considered [Amgoud and Cayrol, 2002; Modgil and Prakken, 2013]. Traditionally, preferences between arguments are used to distinguish an attack from a *defeat* that is known to be a successful attack. The attack from an argument to another one is identified as a defeat if the latter argument is not preferred over the former. In terms of an argumentation graph, the existing literature [Amgoud and Cayrol, 2002; Bench-Capon, 2003; Modgil, 2009; Simari and Loui, 1992] uses this notion to remove unsuccessful attacks from the graph and apply argumentation semantics to the reduced graph. However not all types of attacks are preference-dependent.

The most debated issue in defining preferences between arguments is the distinction between *preference-dependent* and *preference-independent* attacks [Caminada et al., 2014a; Modgil and Prakken, 2013; Prakken, 2012]. Establishing the preference-dependence or independence of attacks requires the explicit representation of the structure of arguments and the nature of attacks between them [Prakken, 2012]. Following [Caminada et al., 2014a; Modgil and Prakken, 2013; Prakken, 2012], we state which types of attacks (i.e. rebuttal, undercut and undermine) require preferences to succeed as defeats.

- Rebuttal: Rebuttal attacks arise from conflicting reasons for and against a conclusion. Therefore, there is no debate that rebuttals are preference-dependent and resolving such attacks needs preferences.

- Undercut: There is consensus on preference-independence of undercuts. A very famous example of undercut attack is Pollock's classic example of an object under the red light [Pollock, 1987]: If an object looks red then it is red. But what if the object is illuminated by red light? Can one still come to the conclusion that the object that looks red is indeed red? Not anymore, since all objects, regardless of their colour, look red under the red light. In this case *a red light is shining* undercuts *if an object looks red then it is red*, as it stops the inference that takes us from *an object looks red* to the conclusion that *the object is red*. The undercut attack here essentially expresses that it is preferred not to draw the inference (e.g. not to come to the conclusion that the object is red) to draw the inference (e.g. to decide that the object is red). This type of preference clearly cannot be captured by defining preferences over arguments [Modgil and Prakken, 2013]. Thus, undercuts are preference-independent.

- Undermine: As for undermine attacks that are attacks to the premise(s) of an argument, preferences are needed with the exception of a premise that makes some assumption in the absence of evidence (e.g. negation as failure in logic programming). Using negation as failure as a premise in an argument (e.g. $not\ \alpha$) makes the argument prone to preference-independent attack from a second arguments that has α as a conclusion, since the former argument relies on α not being provable.

An abstract argumentation framework that explicitly takes argument preferences into account is the Preference-based Argumentation Framework (PAF) that was previously introduced in Chapter 2 (page 35)[1]. Another framework that is capable of representing preferences is Extended Argumentation Framework (EAF) (See Chapter 2, page 36). However, expressing preferences in EAFs requires the preference information to be captured as separate arguments. The preference arguments, when available, determine if an attack is valid by attacking the attack from a less preferred entity to a more preferred one.

In this research we use PAF that allows an explicit representation of preferences. However, there is evidence [Amgoud and Vesic, 2014; Caminada et al., 2014a; Modgil and Prakken, 2013; Prakken, 2012] that PAFs are prone to inconsistency in two situations. We first explain what the situations and resulting inconsistencies are and then explain why these inconsistencies do not occur in our case. The two situations are as follows:

1. When preferences are defined at an abstract level without considering the internal structure of the arguments. The reason for this is that distinguishing between preference-dependent and preference-independent attacks without making the structure of argu-

[1] Value-based Argumentation Framework (VAF) [Bench-Capon, 2002] uses preferences over values instead of arguments (see Chapter 2, Page 35 for more detail).

ments explicit is not possible. For instance, assume that argument a attacks argument b, however it is expressed that argument b is preferred to argument a. At an abstract level, the expressed preference prevents this attack from being identified as a defeat. Now let us see what happens if the the internal structure of arguments were apparent and we knew that a undercuts b. In this case the preference information is irrelevant, since undercuts are not preference-dependent at all. But if we knew a rebuts b, the expressed preference would stop this attack from being identified as defeat. Thus, it is only by the explicit representation of internal structure of arguments that possible inconsistencies can be prevented.

2. When the attack between arguments is not symmetric and preferences are applied, the conflict between arguments may be lost [Modgil and Prakken, 2011], which results in conflicting extensions and violating rationality postulates proposed in [Caminada and Amgoud, 2007]. For instance, assume argument a attacks argument b, however b is preferred to a. In such a case this attack maybe removed and hence the possibility of a and b appearing in the same extension. That said, it is perfectly safe to use preferences in symmetric attacks [Amgoud and Vesic, 2014], since even if one attack is removed from the argumentation graph, there is still one left that ensures capturing the conflict between the two arguments and preventing them from appearing in the same extension.

Regarding the first situation, in the next two sections we make the internal structure of the arguments and the types of attacks (i.e. preference-dependent and preference-independent) between arguments explicit, thus we are not running the risk of encountering inconsistency. As for the second situation, we only define preferences over arguments that symmetrically attack each other. Thus, the use of preferences may reduce a symmetric attack to an asymmetric one, but it cannot lead to complete loss of conflict in any case. Therefore, the second condition cannot be a source of inconsistency in our framework either.

Having established the appropriateness of PAF to argue over plan proposals, we now first give a formal account of PAF, followed by its instantiation with arguments, attacks and preferences required for evaluation of the plan proposals.

Definition 23 (Preference-Based Argumentation Framework (PAF) [Amgoud and Cayrol, 2002; Amgoud and Vesic, 2009, 2014])**.** *A PAF is a tuple of form $\langle Arg, Att, \succeq \rangle$. Arg is a set of arguments, Att and $\succeq$ are attack relations and preference relations between arguments, respectively. A preference relation is a preorder*[2] *on Arg. The preference relation between arguments $Arg_\alpha, Arg_\beta \in Arg$ is therefore denoted as $(Arg_\alpha, Arg_\beta) \in \succeq$. Symbol $\succ$ denotes the strict*[3]

[2]A binary relation is a preorder iff it is reflexive and transitive.
[3]Strict relation is irreflexive and transitive.

relation corresponding to $\succeq$*:* $(Arg_\alpha, Arg_\beta) \in \succ$ *iff* $(Arg_\alpha, Arg_\beta) \in \succeq$ *and* $(Arg_\beta, Arg_\alpha) \notin \succeq$*. Finally,* $(Arg_\alpha, Arg_\beta) \in \sim$ *iff* $(Arg_\alpha, Arg_\beta) \in \succeq$ *and* $(Arg_\beta, Arg_\alpha) \in \succeq$*.*

As discussed earlier, preference information plays a key role in distinguishing attacks from defeats, where the latter is a successful attack. According to Dung [1995] the existence of an attack equals a defeat because all arguments have the same strength and no preferences are defined. Dung therefore maintains that if a attacks b, b is defeated by a. However, in the presence of preferences, an attack is successful if and only if the preference degree or strength of the attacked argument is not greater than the attacker's. An argument a can therefore always attack argument b but it will defeat b if and only if b is not preferred over a [Amgoud and Cayrol, 2002; Delgrande et al., 2004].

Definition 24 (Defeat Relation)**.** *The defeat relation between two arguments* $Def \subseteq Arg \times Arg$*, in a* $PAF = \langle Arg, Att, \succeq\rangle$ *is defined as:* $\forall a, b \in Arg, (a, b) \in Def$ *iff* $(a, b) \in Att$ *and* $(b, a) \notin \succ$*.*

To examine what arguments are justified in an argumentation framework various argumentation semantics have been formulated including the *complete, grounded, preferred, and stable* [Dung, 1995] (See Chapter 2,page 34). These semantics are formulated to examine the justifiability and acceptability of arguments in Dung-style argumentation frameworks [Dung, 1995]. When the preference-based argumentation framework does not suffer from the problems pointed out in page 103 (i.e. when preferences are not applied to asymmetric or preference-independent attacks), mapping a PAF to a Dung's style argumentation framework makes all these semantics available to the former. The following definition proposes such a mapping.

Definition 25 (Mapping of PAF to DAF [Modgil and Bench-Capon, 2011])**.** *A* $PAF = \langle Arg, Att, \succeq\rangle$ *defined in Definition 23 can be mapped to an* $DAF = \langle Arg, Def\rangle$ *using Definition 24. For* $s \in \{$*admissible, complete, preferred, stable, grounded*$\}$*,* E *is an* s *extension of* $PAF = \langle Arg, Att, \succeq\rangle$ *iff* E *is an* s *extension of the Dung framework* $DAF = \langle Arg, Def\rangle$*.*

In order to construct a preference-based argumentation framework that enables the agent to reason about a plan proposal, we need to define the arguments and the way they interact, and the preferences between them. In Chapter 2 (page 36) we surveyed two methods of presenting the internal structure of arguments in an argumentation framework, namely logic-based and scheme-based structures. We also explained that scheme-based argumentation is especially common in computational systems, when arguments need to be structured and formulated diversely to capture the domain-dependent features of the problem that they are modelling

[Atkinson and Bench-Capon, 2007b; Toniolo, 2013; Walton, 1996]. In particular, argumentation schemes have been very popular in practical reasoning since they easily lend themselves to the defeasible nature of reasoning about actions [Atkinson, 2005; Gasque, 2013; Oren, 2013; Toniolo et al., 2012]. Therefore, in this thesis we use argumentation schemes to present the internal structure of arguments. Each scheme consists of a set of premises, a set of conclusions and a set of critical questions associated with it. Critical questions enable challenging an argument built by instantiating the scheme and they can be used for several purposes [Gasque, 2013] including (i) creating or strengthening an argument proposal; (ii) creating arguments and attacks; (iii) challenging an argument put forward by a party; or (iv) rejecting an argument put forward by a party.

We now explain how we are using argumentation schemes and critical questions to reason about a plan proposal. Having all the plans available, the agent reasons about plans by assuming what happens if, for instance, plan π is put forward for investigation. There might be goals that this plan does not satisfy or norms that it violates. Reasoning about plans requires the agent to engage in an internal dialogue and ask itself the following questions: are the goals that are satisfied more important than the goals not satisfied or more important than the norms violated? What about norms that are violated? Are they violated because satisfying a more important goal or complying with a more important norm requires violating the former norm? Such a dialogue thus involves exchanging arguments for plans, goals and norms that are constructed by instantiating three argumentation schemes:

Plan Argument Arg_π: The plan arguments are constructed based on Oren's scheme for a sequence of actions (See AS1 in Figure 2-14, page 50). We construct arguments for any sequence of actions that is identified as a plan.

Goal Argument Arg_g: The scheme based on which the goal arguments are constructed is the Established rules scheme, which was explained in Chapter 2 (page 39). This scheme demands that each of the agent's goals that are feasible should be satisfied in the plan put forward. However, the agent itself may argue against satisfying a goal in a plan if the goal is in conflict with another goal and thus hinders it. Similarly, the agent can argue against satisfying a goal in a plan if the goal achievement hinders complying with a more important norm.

Norm Argument Arg_n: Like goal arguments, norm arguments are built based on the Established rules scheme. Norms are external regulation that are imposed on the agent as the consequences of executing certain actions. Thus, the norms imposed on the agent, depending on the actions executed in a plan, differ from one plan to another. This argument scheme requires the agent to comply with the norms imposed on the agent in a given plan. However, the agent may argue that complying with a certain norm prevents the agent from adhering to a more important norm or satisfying a more important goal in a plan.

To allow questioning arguments that are built based on the schemes, we introduce a set of critical questions for each scheme. These critical questions provide ways to create attacks that challenge and/or reject an argument put forward previously. Figure 5-2 illustrates the interactions between arguments, where nodes represent arguments built based on the schemes and arrows represent the attack relations between arguments defined through the critical questions:

CQ1: Is there any attack from a goal argument to the plan presented by Arg_π?

CQ2: Is there any attack from a norm argument to the plan presented by Arg_π?

CQ3: What goal arguments might attack the goal presented by Arg_g?

CQ4: What norm arguments might attack the goal presented by Arg_g? or What goal arguments might attack the norm presented by Arg_n?

CQ5: What norm arguments might attack the norm presented by Arg_n?

The purpose of CQ1 and CQ2 is to provide ways to challenge a plan argument, CQ3 to challenge a goal argument, CQ4 to challenge a goal or norm argument, and CQ5 to challenge a norm argument. The three argument schemes and their associated critical questions will be formally defined in the next two sections.

Argument Schemes and Argument Construction

In the previous section we mentioned three argument schemes in order to construct a set of arguments for normative practical reasoning: plan arguments, goal arguments and norm arguments. This section gives the formal account of each scheme.

Plan Arguments This argument scheme results in constructing an argument for each plan obtained from the implementation of our formal model. The scheme for plans is inspired by Oren's scheme [Oren, 2013] for a sequence of actions (AS1 in Figure 2-14, page 50) – "AS1: In situation S, the sequence of joint actions $A_1, \cdots, A_n$ should be executed." – and Atkinson's scheme for plans in BDI agents [Atkinson, 2005, p. 95] – "Given the current situation R, there is a plan A which if performed will bring about S, realising G which promotes V".

Definition 26 (Plan Argument Arg_π)**.** *A plan argument is used to claim that the agent should execute the proposed sequence because the sequence of actions leads to satisfying a set of goals G_π, and complying with a set of norms $N_{cmp(\pi)}$, although it violates some norms $N_{vol(\pi)}$.*

- In the initial state Δ
- The agent should execute sequence of actions $\pi = \langle (a_0, 0), \cdots, (a_n, t_{a_n}) \rangle$
- which satisfies a set of goals G_π, complies with a set of norms $N_{cmp(\pi)}$ and violates a set of norms $N_{vol(\pi)}$

Equation 5.1 shows a formalisation of this scheme based on our formal model:

$$Arg_\Pi = \{Arg_\pi \text{ s.t. } \pi \in \Pi\} \tag{5.1}$$

Goal Arguments This argument scheme results in constructing an argument for each goal that is feasible. A goal argument is used to explore why a goal is not satisfied in a plan, or to address the conflict between two goals or a goal and a norm. In Chapter 2 (page 24) we informally defined what it means for a goal to be feasible, namely being satisfied in at least one plan. We also discussed that if there is no plan to satisfy a goal, a rational agent should not adopt that goal or try to justify its adoption since it is not feasible to begin with. Goal arguments are therefore only constructed for feasible goals. We first recall from Chapter 2 (page 39) Walton's Established rules scheme [Walton et al., 2008] on which the argument scheme for goals is based:

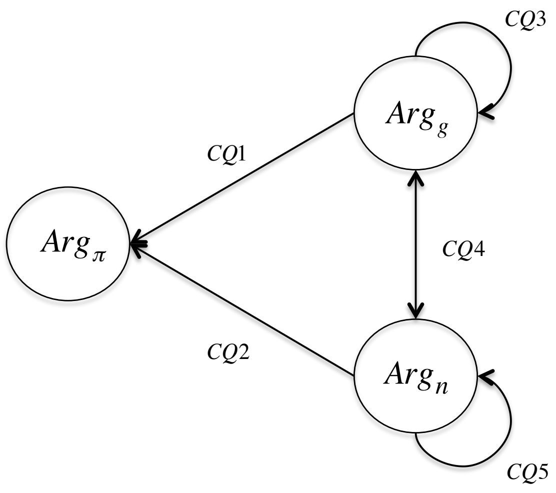


Figure 5-2: Interaction between Arguments

- If A is the case, then an evaluation E is justified/ conduct C is required.
- A is the case.
- Therefore, evaluation E is justified/ conduct C is required.

In order to formulate this argument scheme, we first provide a formal account of *goal feasibility*.

Definition 27 (Goal Feasibility)**.** *A goal is feasible if it is satisfied in at least one plan:*

$$G_{fsb} = \{g \in G \textit{ s.t. } \exists \pi \in \Pi, \pi \models g\}$$

Definition 28 (Goal Argument Arg_g)**.** *A goal argument claims that a feasible goal should be satisfied:*

- *Goal g is a feasible goal of the agent*
- *Therefore, satisfying g is required*

Equation 5.2 shows a formalisation of this scheme based on Definition 27.

$$Arg_G = \{Arg_g \text{ s.t. } g \in G_{fsb}\} \tag{5.2}$$

Norm Arguments This argument scheme results in constructing an argument for each norm that is activated in the plan proposal. A norm argument is used to explore why a norm is violated in a plan. It is also used to address the conflict between two norms or a goal and a norm. Similar to goal arguments, norm arguments are based on the Established rules scheme. Before formulating this scheme, we repeat that depending on the actions executed in a plan, a norm is not necessarily activated in all plans. The set of activated norms in plan π were denoted as N_π in Chapter 3 (page 65).

Definition 29 (Norm Argument Arg_n)**.** *A norm argument claims that an activated norm should be complied with:*

- *Norm n is an activated norm imposed on the agent in plan π*
- *Therefore, complying with n is required in π*

Equation 5.3 shows a formalisation of this scheme based on Definition 14.

$$Arg_{N_\pi} = \{Arg_n \text{ s.t. } n \in N_\pi\} \tag{5.3}$$

Critical Questions and Interactions between Arguments

In this section we demonstrate how arguments built using schemes in the previous section attack each other using critical questions mentioned earlier (page 107). We also mention the types of attacks caused by each critical question.

Critical Questions Associated with Argument Scheme for Plans

CQ1: Is there any attack from a goal argument to the plan presented by Arg_π?

This CQ results in an undercut attack from a goal argument to a plan argument, when the goal is not satisfied in the plan. Undercut attacks caused by CQ1 are by definition asymmetric and are formulated as:

$$\forall Arg_g \in Arg_G, Arg_\pi \in Arg_\Pi \text{ if } g \notin G_\pi \text{ then } (Arg_g, Arg_\pi) \in Att \tag{5.4}$$

CQ2: Is there any attack from a norm argument to the plan presented by Arg_π?

This CQ results in an undercut attack from a norm argument to a plan argument, when the norm is violated in the plan. This asymmetric attack is formulated as:

$$\forall Arg_n \in Arg_{N_\pi}, Arg_\pi \in Arg_\Pi \text{ if } n \in N_{vol(\pi)} \text{ then } (Arg_n, Arg_\pi) \in Att \tag{5.5}$$

Critical Questions Associated with Argument Scheme for Goals

CQ3: What goal arguments might attack the goal presented by Arg_g?

This CQ results in a rebut attack between arguments for conflicting goals. Two goals are in conflict if satisfying one requires bringing about a state of affairs that is in conflict with the state of affairs required for satisfying the other (see Chapter 3, page 67). Attacks caused by CQ3 are by definition symmetric and irreflexive. This can be formulated as:

$$\forall Arg_g, Arg_{g'} \in Arg_G \text{ if } (g, g') \in cf_{goal} \text{ then } (Arg_g, Arg_{g'}) \in Att \tag{5.6}$$

CQ4: What norm arguments might attack the goal presented by Arg_g?

The conflict between a norm and a goal was defined in Chapter 3 (page 69) as follows: an obligation norm and a goal are in conflict, if executing the action that is the subject of the obligation brings about postconditions that are in conflict with the requirements of the goal; and a prohibition norm and a goal are in conflict, if the postconditions of the action that is the subject of prohibition contribute to satisfying the goal, but the action execution is prohibited by the norm. Rebut attacks caused by CQ4 are by definition symmetric and are formulated below.

$$\forall Arg_g \in Arg_G, Arg_n \in Arg_{N_\pi} \text{ if } (g, n) \in cf_{goalnorm} \text{ then } (Arg_g, Arg_n) \in Att \tag{5.7}$$

Critical Questions Associated with Argument Scheme for Norms

CQ4: Whatgoalarguments might attack the norm presented by Arg_n?

The previous critical question, is associated with argument schemes for norms as well as goals, hence the repetition of the number of critical question. As addressed in the previous critical question, the conflict between a norm and a goal is symmetric. Therefore, if a goal and a norm are in conflict, the goal argument attacks the norm argument and the other way around.

$$\forall Arg_g \in Arg_G, Arg_n \in Arg_{N_\pi} \text{ if } (n, g) \in cf_{goalnorm} \text{ then } (Arg_n, Arg_g) \in Att \tag{5.8}$$

CQ5: What norm arguments might attack the norm presented by Arg_n?

Conflict between two norms was defined as a contextual conflict that depends upon the context of the plan in which the norms are activated (See Chapter 3, page 70). Two obligations are in conflict in the context of a plan if the obliged actions have a concurrency conflict, yet they are required to be executed concurrently. Moreover, an obligation and a prohibition are in conflict if the prohibition forbids the agent to execute the action that is the subject of the obligation. Similar to CQ3 and CQ4, rebut attacks caused by CQ5 are symmetric and irreflexive:

$$\forall Arg_n, Arg_{n'} \in Arg_{N_\pi} \text{ if } (n, n') \in cf^{\pi}_{norm} \text{ then } (Arg_n, Arg_{n'}) \in Att \tag{5.9}$$

Preference Relation between Arguments

In the previous section we defined five critical questions that address the attack between arguments. We also defined the types of attacks caused by each critical question. Table 5.1 shows the need for preferences for resolving attacks caused by each critical question. As it is evident, attacks due to CQ3, CQ4 and CQ5 are rebuttals and therefore need preferences to resolve. In contrast, attacks caused by CQ1 and CQ2 do not need preferences.

In Chapter 2 (page 24) we made a comparison between quantitative and qualitative representation of preference information. We, in particular, referred to reasons enumerated by Doyle

Critical Question	Entities Involved	Type of Attack	Preference-dependence
CQ1	goal-plan	undercut	preference-independent
CQ2	norm-plan	undercut	preference-independent
CQ3	goal-goal	rebut	preference-dependent
CQ4	goal-norm/norm-goal	rebut	preference-dependent
CQ5	norm-norm	rebut	preference-dependent

Table 5.1: Preferences between Arguments

and Thomason [1999]; Fox and Parsons [1998] and Prakken [2006a] to justify why we choose a qualitative preference ordering to express agent preferences. The impracticality of generating a set of probabilities and utilities required by quantitative methods and their adequacy in capturing realistic cases (e.g. generic preferences that are common human expressions), were among the most important of these justifications. We now describe how the agent preferences can be formulated and consequently translated to preferences between arguments. Note that goal-based agents that constantly prefer their goals to norms and norm-based agents that always prefer their norms to their goals can both be modelled using this definition.

Definition 30 (Preference between Goals and Norms)**.** *We define $\succeq^{gn}$ as a partial preorder on $G \cup N$. The preference relation between $\alpha, \beta \in G \cup N$ is therefore denoted as $(\alpha, \beta) \in \succeq^{gn}$ and reads as satisfying goal α (or complying with norm α) is at least as preferred as satisfying goal β (or complying with norm β). Symbol $\succ^{gn}$ denotes the strict relation corresponding to $\succeq^{gn}$: $(\alpha, \beta) \in \succ^{gn}$ iff $(\alpha, \beta) \in \succeq^{gn}$ and $(\beta, \alpha) \notin \succeq^{gn}$. $(\alpha, \beta) \in \sim^{gn}$ iff $(\alpha, \beta) \in \succeq^{gn}$ and $(\beta, \alpha) \in \succeq^{gn}$.*

Definition 31 summarises the mapping from agent preferences to argument preferences as the following. Note that since each argument represents a single goal or norm, the mapping, cannot lead to inconsistencies in the preference ordering for a single argument. In structured formalisms that arguments are a chain of sub-arguments, additional consideration is required to ensure the rationality of preferences in a single argument [Brewka and Eiter, 2000].

Definition 31 (Preference Relation between Arguments)**.** *The preference relation between arguments is denoted as $\succeq$: $(Arg_\alpha, Arg_\beta) \in \succeq$ iff $(\alpha, \beta) \in \succeq^{gn}$; $(Arg_\alpha, Arg_\beta) \in \succ$ iff $(\alpha, \beta) \in \succ^{gn}$; $(Arg_\alpha, Arg_\beta) \in \sim$ iff $(\alpha, \beta) \in \sim^{gn}$; $(Arg_\alpha, Arg_\beta) \in \succeq$ reads as argument Arg_α is at least as strong as argument Arg_β, while $(Arg_\alpha, Arg_\beta) \in \succ$ reads as Arg_α is stronger than argument Arg_β.*

Having defined the three elements of a PAF for plan proposals (arguments, attacks and preferences), an instantiated version of the PAF in Definition 23 for a plan proposal is given in

Definition 32. An argumentation framework for a plan proposal consists of the plan argument itself, goal arguments, and norm arguments that are activated in this plan. The set of attacks between arguments is expressed through critical question CQ1–CQ5 and the preference relation is defined as in Definition 31, where the preferences are used to map attacks to defeats when possible.

Definition 32 (Plan Proposal Argumentation Framework)**.** *A* $DAF_\pi = \langle Arg, Def \rangle$*, where* $Arg = Arg_\pi \cup Arg_G \cup Arg_{N_\pi}$ *and* $Att = Att_{CQ1-5}$*. The defeat relation* Def *is defined as:* $\forall Arg_\alpha, Arg_\beta \in Arg, (Arg_\alpha, Arg_\beta) \in Def$ *iff* $(Arg_\alpha, Arg_\beta) \in Att$ *and* $(Arg_\beta, Arg_\alpha) \notin \succ$*.*

In this section, we built an argumentation framework for a plan proposal with the aim of checking the justifiability of the plan proposal. In the next section we discuss how (i.e. based on which semantics) to check the justifiability of plan proposals.

5.1.2 Evaluation of the Argumentation Framework

In Chapter 2 (page 34) we talked about argumentation semantics as means to evaluate argumentation frameworks. In deciding what semantics (i.e. what type of evaluation) one should use, the purpose of the evaluation plays a key role. Although various semantics have been introduced since the proposal of Dung's argumentation framework [Dung, 1995], they have mainly been defined in terms of mathematical formulation. However, it is argued [Caminada, 2010; Caminada et al., 2014b; Prakken, 2006a] that in order to make these semantics more accessible, in addition to mathematical formulation, the link between semantics and the type of reasoning they are associated with should have greater emphasis. In Chapter 2 (page 20) we distinguished between reasoning about beliefs, namely epistemic reasoning, and reasoning about actions, namely practical reasoning. The question we are trying to debate in this section is *Having formed an argumentation framework about practical attitudes of the agent (i.e. plans, goals, and norms) what semantics should be used for the evaluation of this framework, such that the agent is sure to make justified decisions?* To make it more general *What semantic(s) is suitable for practical reasoning?*

The most crucial point in answering this question is how conflicts are to be handled. Arguments are defeasible rules and potential conflicts between defeasible conclusions can be handled differently. The most prominent types of inference to deal with conflicts are *credulous and sceptical* reasoning. The former treats arguments in such a conflict as alternatives, yielding to multiple defeasible conclusions, whereas, the latter discards both of the conclusions because of their unresolvable conflict. Going back to the question of what semantic is suitable for practical reasoning, it is necessary to pinpoint whether reasoning about and toward actions has to be credulous or sceptical.

With the exception of [Amgoud, 2012], the rest of the works [Broersen et al., 2002; Oren, 2013; Prakken, 2006a; Thomason, 2000] in the field of argumentation-based practical reasoning and decision-making are unanimous that reasoning about and toward actions has to be credulous. An agent can have an inconsistent set of actions, goals or desires [Prakken, 2006a; Searle, 2001]. If the conflict between goals for example is unresolvable then what we want is to have choices between them rather than rejecting both of them. The philosophical and pragmatic foundation of credulous inference for practical reasoning were briefly mentioned in Chapter 2 (page 46) and are discussed in detail by Caminada [2006] and Prakken [2006a]. The main argument for this claim is that credulous inference preserve choices and produces multiple alternatives in the case of unresolvable conflict between arguments, whereas sceptical inference rejects both arguments in an unresolvable conflict. Caminada [2006] explains this argument in terms of *argument labellings*. In order to provide a more precise version of his argument we first define Caminada's Complete labellings.

Definition 33 (Complete Labellings [Caminada, 2006]). *Let $\langle Arg, Def\rangle$ be an argumentation framework, an AF-labelling is a total function $\mathcal{L} : Arg \rightarrow \{in, out, undec\}$. $in(\mathcal{L})$ is defined as $\{a \in Arg$ s.t. $\mathcal{L}(a) = in\}, out(\mathcal{L})$ as $\{a \in Arg$ s.t. $\mathcal{L}(a) = out\}$, and $undec(\mathcal{L})$ as $\{A \in Arg$ s.t. $\mathcal{L}(a) = undec\}$. A non-partial argument labelling is called a complete labelling, $\mathcal{L}_{cmp}$, iff for each argument $a \in Arg$ it holds that a is labelled in iff each attacker of a is labelled out and a is labelled out iff there exists an attacker of a that is labelled in.*

Complete labellings can be seen as subjective but reasonable point of view that an agent can take with respect to which arguments are in, out or $undec$. The set of all possible complete labellings therefore stands for all possible justified positions that an agent can take. From a labelling perspective, an argument is justified under sceptical inference if it is in *every* reasonable position and is justified under credulous inference if it is in *at least one* reasonable position. Thus, what the agent needs when reasoning about its practical attitudes is to see whether they are at least in one justified position out of all possible justified positions. Being in at least one justified position coincides with being labelled in by at least one complete labelling, or being credulously accepted under complete semantics [Caminada, 2006]. Also it is a known fact that the credulous preferred semantics coincides with the credulous complete semantics [Caminada, 2006]. Consequently, Broersen et al. [2002]; Oren [2013]; Prakken [2006a]; Thomason [2000] not only proposed credulous reasoning about and toward actions, but more precisely they prescribe credulous preferred semantics for practical reasoning purposes. We therefore use the intuition behind credulous preferred semantics to evaluate the argumentation framework built for each plan proposal. The goal of this evaluation is to determine what plan arguments and essentially what plans are justified with respect to the context of the plan and the preference information at hand. Definition 34 defines preferred semantics in terms of labellings, which will be used in defining the justified plans (Definition 35).

Definition 34 (Preferred Labellings [Caminada, 2006])**.** *A complete labelling is called a preferred labelling, $\mathcal{L}_{pr}$, iff its set of in-labelled arguments is maximal (or equivalently, iff its set of out-labelled arguments is maximal) with respect to set inclusion.*

Definition 35 (Justified Plan)**.** *Plan π is justified if Arg_π is labelled in by at least one preferred labelling for DAF_π: $\exists\, \mathcal{L}_{pr}$ s.t. $Arg_\pi \in in(\mathcal{L}_{pr})$.*

Properties of justified plans are investigated in the next section.

5.1.3 Properties of Justified Plans

We know for a fact that every preferred labelling is a complete labelling. An argument is labelled in by a complete labelling iff all its attackers are labelled out. Therefore, a plan argument is labelled in by a preferred labelling iff all its attackers are labelled out by that labelling. The attackers of a plan argument are the arguments for goals that it does not satisfy, denoted as $Arg_{G\setminus G_\pi} = \{Arg_g \text{ s.t. } g \in G \setminus G_\pi\}$, and the arguments for norms that it violates, denoted as $Arg_{N_{vol(\pi)}} = \{Arg_n \text{ s.t. } n \in N_{vol(\pi)}\}$. These attacks are caused by CQ1 and CQ2 which are both preference-independent and therefore they will always succeed as a defeat. So we have:

Property 5.1. $Arg_\pi \in in(\mathcal{L}_{pr}) \Leftrightarrow Arg_{G\setminus G_\pi} \cup Arg_{N_{vol(\pi)}} \subseteq out(\mathcal{L}_{pr})$

The intuition behind the above property is that if a plan is justified, the goals that it does not satisfy and norms that it violates cannot be justified. In contrast, in property 5.2 we show that if a plan is justified, the goals and norms that it satisfies and complies with are also justified.

Property 5.2. $Arg_\pi \in in(\mathcal{L}_{pr}) \Rightarrow Arg_{G_\pi} \cup Arg_{N_{cmp(\pi)}} \subseteq in(\mathcal{L}_{pr})$

Proof. Since $Arg_\pi \in in(\mathcal{L}_{pr})$, from Property 5.1 we know that $Arg_{G\setminus G_\pi} \cup Arg_{N_{vol(\pi)}} \subseteq out(\mathcal{L}_{pr})$. We also know from the definition of a plan that $Arg_{G_\pi} \cup Arg_{N_{cmp(\pi)}}$ is conflict free. Since all possible attackers of $Arg_{G_\pi} \cup Arg_{N_{cmp(\pi)}}$ belong to $Arg_{G\setminus G_\pi} \cup Arg_{N_{vol(\pi)}}$ and $Arg_{G\setminus G_\pi} \cup Arg_{N_{vol(\pi)}}$ are all labelled out, we conclude that $Arg_{G_\pi} \cup Arg_{N_{cmp(\pi)}} \subseteq in(\mathcal{L}_{pr})$.

Note that from $Arg_{G_\pi} \cup Arg_{N_{cmp(\pi)}} \subseteq in(\mathcal{L}_{pr})$ one cannot necessarily conclude that $Arg_\pi \in in(\mathcal{L}_{pr})$. Figure 5-3 shows a plan that satisfies goal g_1. Although $Arg_{G_\pi} = \{Arg_{g_1}\} \subseteq in(\mathcal{L}_{pr})$, $Arg_\pi \notin in(\mathcal{L}_{pr})$.

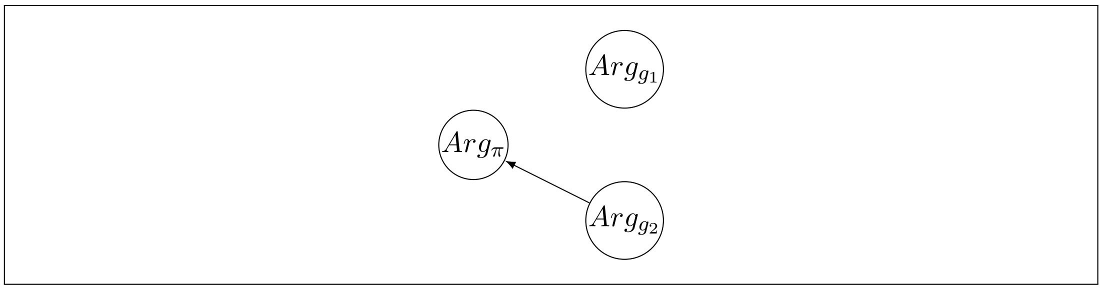


Figure 5-3

It is provable that there is either none or exactly one preferred labelling that labels Arg_{π} in, which means there is (if any) only a single position that the agent can take to justify Arg_{π}, and therefore, plan π:

Property 5.3. *There is no more than one preferred labelling in which $Arg_{\pi} \in in(\mathcal{L}_{pr})$ (i.e. plan π is justified).*

Proof. From Property 5.1 and 5.2 we know that if $Arg_{\pi} \in in(\mathcal{L}_{pr})$ then $Arg_{G \setminus G_{\pi}} \cup Arg_{N_{vol(\pi)}} \subseteq out(\mathcal{L}_{pr})$ and $Arg_{G_{\pi}} \cup Arg_{N_{cmp(\pi)}} \subseteq in(\mathcal{L}_{pr})$. Since every preferred labelling is a complete labelling and the following property [Caminada and Gabbay, 2009] holds for complete labellings $\mathcal{L}_{cmp1}$ and $\mathcal{L}_{cmp2}$: if $out(\mathcal{L}_{cmp1}) = out(\mathcal{L}_{cmp2})$ then $\mathcal{L}_{cmp1} = \mathcal{L}_{cmp2}$; we conclude that there is no more than one preferred labelling in which $Arg_{\pi} \in in(\mathcal{L}_{pr})$

Property 5.4. *If $Arg_{\pi} \in in(\mathcal{L}_{pr})$, $\mathcal{L}_{pr}$ is a stable labelling.*

Proof. In Property 5.1 and 5.2 we showed that if $Arg_{\pi} \in in(\mathcal{L}_{pr})$ then $Arg_{G \setminus G_{\pi}} \cup Arg_{N_{vol(\pi)}} \subseteq out(\mathcal{L}_{pr})$ and $Arg_{G_{\pi}} \cup Arg_{N_{cmp(\pi)}} \subseteq in(\mathcal{L}_{pr})$, which makes the $undec(\mathcal{L}_{pr}) = \emptyset$. A complete labelling with $undec(\mathcal{L}) = \emptyset$ is a stable labelling. Since $\mathcal{L}_{pr}$ is a preferred labelling and every preferred labelling, is a complete labelling, essentially $\mathcal{L}_{pr}$ is a complete labelling in which $undec = \emptyset$. Therefore, $\mathcal{L}_{pr}$ is a stable labelling.

The above property can be generalised further. In fact all preferred labellings of an argumentation framework for a plan proposal are stable labellings and that is regardless of Arg_{π} belonging to the preferred labellings. The reason behind this generalisation is that an argumentation framework for a plan proposal cannot contain odd-length cycle of arguments and therefore it cannot contain any argument labelled as $undec$ under the preferred semantics [Baroni and Giacomin, 2003; Bodanza and Tohmé, 2009; Prakken and Vreeswijk, 2002]. Thus, all preferred labellings are also stable labellings.

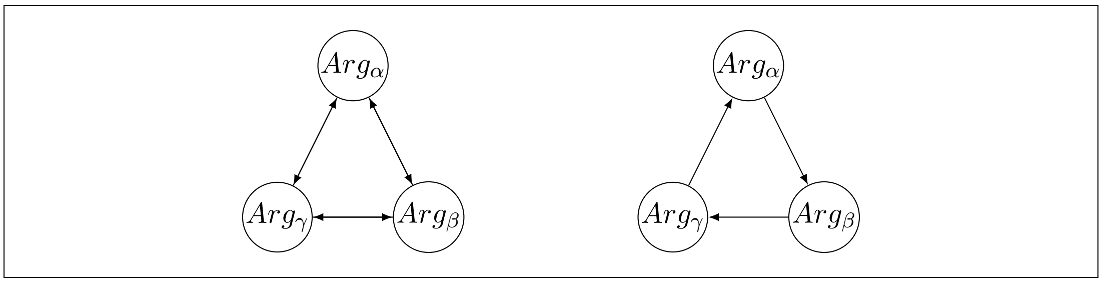


Figure 5-4: Odd-length Cycle

A plan argument can never be a part of an odd-length cycle, since it does not attack any argument. So if there is any odd-length cycle in the framework, it is formed by goal and norm arguments. The attack within and between goal and norm arguments is symmetric (before applying preferences). Figure 5-4 (left-hand side) shows a cycle between three goal/norm arguments. Is it possible to get from the graph on the left-hand side to the odd-length cycle on the right-hand side? The two-way attack between Arg_α and Arg_β is reduced to a single defeat from Arg_α and Arg_β if $(Arg_\alpha, Arg_\beta) \in \succ$, which means that $(\alpha, \beta) \in \succ^{gn}$. $(\alpha, \beta) \in \succ^{gn}$ if and only if $(\alpha, \beta) \in \succeq^{gn}$ and $(\beta, \alpha) \notin \succeq^{gn}$. The same applies to the two-way attack between Arg_β and Arg_γ. Thus, we must have had $(\beta, \gamma) \in \succeq^{gn}$ and $(\gamma, \beta) \notin \succeq^{gn}$. Since $\succeq^{gn}$ is a preorder and hence transitive, from $(\alpha, \beta) \in \succ^{gn}$ and $(\beta, \gamma) \in \succeq^{gn}$ we conclude that $(\alpha, \gamma) \in \succeq^{gn}$. Now, either $(\gamma, \alpha) \in \succeq^{gn}$, which according to Definition 30 means that $(\alpha, \gamma) \in \sim^{gn}$ and the symmetric attack between Arg_α and Arg_γ cannot be reduced to a single defeat from Arg_γ to Arg_α; or $(\gamma, \alpha) \notin \succeq^{gn}$, which means that he symmetric attack between Arg_α and Arg_γ is reduced to a single defeat from Arg_α to Arg_γ. In either case, obtaining the odd-length cycle on the right-hand side of Figure 5-4 is impossible.

5.2 Best Plans

In the previous sections, we described the process of building an argumentation framework for a plan proposal to check the justifiability of plans as a step toward identifying the best plan. We also established that justified plans are defined as credulously accepted under the preferred semantics. Although all the justified plans are internally coherent and defendable by the agent, there could be further criteria that makes the agent disagree with one or be in better agreement with another one. The issue of what plan is the best plan out of a set of justified plans has been treated differently by different scholars. Some works [Amgoud et al., 2008a] do not distinguish between justified plans and take them as “as good as” each other. Therefore, all the justified plans are essentially the best plans. Simply maximising the number of achieved desires is the basis of comparison of justified options in Hulstijn and van der Torre [2004]. Rahwan and Amgoud [2006] use the utility of plans (i.e. the worth of desires and the cost of resources to achieve them) to find the best plan out of the justified ones. That is of course subject to the availability of the exact measurement of utility.

In this work, what is available to the agent is a partial preference order over goals and norms. In the previous section we used preferences for handling attacks and defeats. Recently Amgoud and Vesic [2014] has explored the role of preferences to refine the evaluation of arguments by using preferences to choose some extensions among a set of extensions under a specific semantics. Since each extension is essentially a set of arguments, preference-based set ordering techniques are used to compare extensions. A comprehensive survey of methods for ranking sets of objects can be found in [Barber et al., 2001]. One of the set ordering principles that has received a lot of attention in the argumentation community is the *Democratic* set ordering principle. Essentially, democratic ordering ranks set S_i above set S_j if any object in S_j is weaker than at least one object in set S_i. Democratic ordering was first used in argumentation frameworks by Prakken and Sartor [1997] and more recently in [Amgoud and Vesic, 2014; Caminada et al., 2014a]. Prakken and Sartor [1997] and Caminada et al. [2014a] use democratic ordering to compare two arguments made of a set of defeasible rules based on the preferences between defeasible rules in the two arguments. Amgoud and Vesic [2014], however, use democratic ordering to discard some extensions of an argumentation framework under a certain semantics.

We use the democratic ordering for further refinement of justified plans by considering the combination of goals satisfied and norms violated in justified plans. We have chosen the democratic principle due to its effectiveness in handling even-length cycles of arguments. Since the preferences over goals and norms is partial, comparing two plans based on democratic ordering is not always possible. Therefore, in absence of such preference information, the best plan is defined as a plan that satisfies most goals while violating the fewest norms. We first give a formal account of democratic ordering and then define the *goal-dominant* and *norm-dominant* plans, based on which a *better than* relation between plans is defined. We also give an example of when democratic ordering can be useful in our framework.

Definition 36 (Democratic Ordering). *Let S_i and S_j be two sets of objects. According to democratic ordering (denoted as $\trianglerighteq$) $(S_i, S_j) \in \trianglerighteq$ iff $\forall \beta \in S_j \backslash S_i, \exists \alpha \in S_i \backslash S_j$ s.t. $(\alpha, \beta) \in \succ$. As usual $(S_i, S_j) \in \triangleright$ iff $(S_i, S_j) \in \trianglerighteq$ and $(S_i, S_j) \notin \trianglerighteq$.*

Democratic ordering is reflexive and transitive[4].

[4]See Amgoud and Vesic [2014] for proof.

Definition 37 (Goal Dominance: Democratic Ordering)**.** *Let G_{π_i} and G_{π_j} be the sets of goals satisfied in plan π_i and π_j. According to democratic ordering $(G_{\pi_i}, G_{\pi_j}) \in \trianglerighteq_G$ iff $\forall g' \in G_{\pi_j} \setminus G_{\pi_i}, \exists g \in G_{\pi_i} \setminus G_{\pi_j}$ s.t. $(g, g') \in \succ^{gn}$.*

Let $\succ^g$ be the ordering on G induced by $\succ^{gn}$.

Theorem 5.1. *$\trianglerighteq_G$ is a total preorder on $P(G)$[5] iff $\succ^g$ is total.*

Proof. Assume that $\trianglerighteq_G$ is a total preorder on $P(G)$, while $\succ^g$ is not a total order on G. The latter means that $\exists g, g' \in G$ s.t.$(g, g') \notin \succ^g$ and $(g', g) \notin \succ^g$. Since $\{g\}$ and $\{g'\}$ both belong to $P(G)$ and $(g, g') \notin \succ^g$ and $(g', g) \notin \succ^g$, we conclude that $\trianglerighteq_G$ is not a total preorder which is contrary to assumption.

Assume that $\succ^g$ is a total order on G. In oder to prove that $\trianglerighteq_G$ is a total preorder, we need to prove that it is reflexive, transitive, and total. It is easy to see it is reflexive and transitive (see Definition 36). Below we prove that it is total meaning that $\forall G_i, G_j \in P(G)$ either $(G_i, G_j) \in \trianglerighteq_G$ or $(G_j, G_i) \in \trianglerighteq_G$. Let $g = max((G_i \cup G_j) \setminus (G_i \cap G_j))$:

case 1: $g \in G_i$. Then $\forall g' \in G_j, \exists g \in G_i$ s.t. $(g, g') \in \succ^g$, which means that $(G_i, G_j) \in \trianglerighteq_G$.

case 2: $g \in G_j$. Then $\forall g' \in G_i, \exists g \in G_j$ s.t. $(g, g') \in \succ^g$, which means that $(G_j, G_i) \in \trianglerighteq_G$.

Now let us see when the relation defined in Definition 37 is a total preorder on $G_\Pi = \{G_{\pi_1}, G_{\pi_2}, \cdots, G_{\pi_n}\}$, where as before, G_{π_i} is the set of goals satisfied in plan i .

Corollary 5.1. *$\trianglerighteq_G$ is a total preorder on G_Π if $\succ^g$ is total.*

The above corollary follows immediately from Theorem 5.1 because $G_\Pi \subseteq P(G)$. However, note that having a total preorder on $\trianglerighteq_G$, dose not necessarily mean we need to have a total order on G. Here is a counter example.

Example 5.1. *Let $G = \{g_1, g_2, g_3\}$ and $\succ^g = \{(g_2, g_3)\}$. Also let $G_\Pi = \{G_{\pi_1}, G_{\pi_2}\}$, where $G_{\pi_1} = \{g_1, g_2\}$ and $G_{\pi_2} = \{g_1, g_3\}$. It is clear that $(G_{\pi_1}, G_{\pi_2}) \in \trianglerighteq_G$, which essentially means we have a total preorder on G_Π while $\succ^g$ is not a total order on G.*

Plan π_i goal-dominates plan π_j, if for every goal satisfied in π_j that is not satisfied in π_i, there is at least one preferred goal satisfied in π_i that is not satisfied in π_j. If the preference information is not sufficient to conduct such a comparison, we define goal dominance as satisfying a greater number of goals.

[5] $P(G)$ is the power set of G.

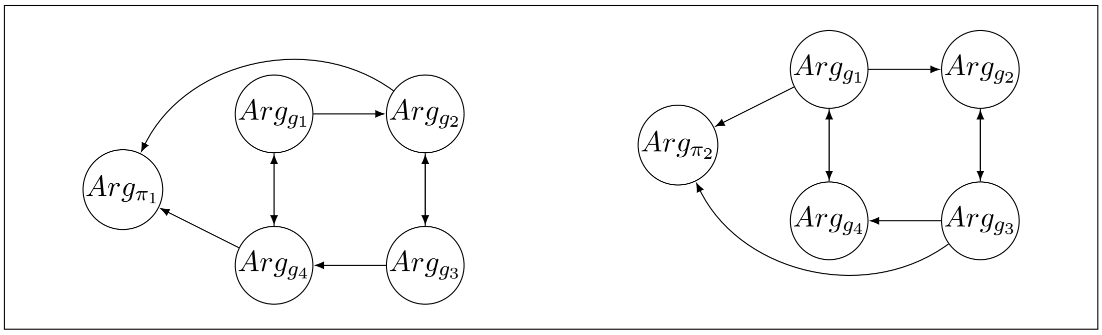


Figure 5-5: Argumentation Graphs for Plans π_1 and π_2

Definition 38 (Goal-dominance). *Plan π_i goal-dominates π_j denoted as $(\pi_i, \pi_j) \in \geq_G$*

1. *if $\trianglerighteq_G$ is a total preorder on G_Π and $(G_{\pi_i}, G_{\pi_j}) \in \trianglerighteq_G$; Else (i.e. if $\trianglerighteq_G$ is not a total preorder on G_Π)*
2. *if $|G_{\pi_i}| \geq |G_{\pi_j}|$.*

$>_G$ is the strict relation associated with $\geq_G$. $(\pi_i, \pi_j) \in \sim_G$ iff $(\pi_j, \pi_i) \in \geq_G$ and $(\pi_i, \pi_j) \in \geq_G$.

It is straight forward to see $>_G$ is irreflexive, antisymmetric and transitive, while $\sim_G$ is reflexive, symmetric and transitive.

Example 5.2. *Assume an agent with four goals g_1, g_2, g_3 and g_4, such that $\{(g_1, g_2), (g_2, g_3), (g_3, g_4), (g_4, g_1)\} \in cf_{goal}$. Let π_1 and π_2 be two plans for the the agent such that $G_{\pi_1} = \{g_1, g_3\}$ and $G_{\pi_2} = \{g_2, g_4\}$. Also the agent prefers satisfying g_1 to satisfying g2, and g_3 to g_4. Figures 5-5 show the argumentation graph associated with these plans. Both plans are justified, however according to democratic principle $(\pi_1, \pi_2) \in \geq_G$ since $(g_1, g_2) \in \succ^{gn}$ and $(g_3, g_4) \in \succ^{gn}$. Note that if we defined goal-dominance only based on the number of goals that are satisfied in two plans, both plans would have been equally good, neglecting the preferences between goals.*

In order to define norm dominance we look at the normative quality of the plans. Although norm compliance mechanisms abstain from regimenting norms since that would eliminate the possibility of violation, they have used different mechanisms to discourage the agent from norm violation, such as (i) associating violation to loss of utility; (ii) negatively influencing the agent reputation; and (iii) hindering the agent's individual goals. Avoiding violation in normative practical reasoning and planning is manifested in [Oren, 2013] by expressing that not violating a norm (i.e. complying or avoiding) is preferred over violating it. Also Oren et al. [2013] induces preferences over paths obtained from a transition system based on the fact that fewer number of violations are always preferred. However, in their system it is possible for a norm to be activated several times in the same path. Thus, instead of comparing the set of violated norms in two plans, they compare the number of times that a specific norm is violated in two paths. Going back to the design choices we made in page 97, Chapter 4, an action cannot be executed more than once in the same plan. Since, we merely focus on modelling action-based norms, the constraint on the occurrence of actions also means that the action that is the activation condition of a norm cannot be executed more than once. Consequently, a norm can at most be activated once in a plan. As a result, instead of talking about the number of times that a single norm is violated in a plan, we are interested in the set of norms that are violated in a plan. Thus, similar to Definition 38, norm dominance is defined based on democratic ordering and number of violations as: Plan π_i norm-dominates plan π_j, if for every norm violated in π_j that is not violated in π_i, there is at least one preferred norm violated in π_i that is not violated in π_j.

Definition 39 (Norm Dominance: Democratic Ordering)**.** *Let $N_{vol(\pi_i)}$ and $N_{vol(\pi_j)}$ be the sets of norms violated in plan π_i and π_j. According to democratic ordering $(N_{vol(\pi_i)}, N_{vol(\pi_j)}) \in \trianglerighteq_N$ iff $\forall n' \in N_{vol(\pi_j)} \setminus N_{vol(\pi_i)}, \exists n \in N_{vol(\pi_i)} \setminus N_{vol(\pi_j)}$ s.t. $(n, n') \in \succ^{gn}$.*

Let $\succ^g$ be the ordering on G induced by $\succ^{gn}$.

Let $N_{vol(\Pi)} = \{N_{vol(\pi_1)}, N_{vol(\pi_2)}, \cdots, N_{vol(\pi_n)}\}$, where as before, $N_{vol(\pi_i)}$ is the set of norms violated in plan i .

Definition 40 (Norm-dominance)**.** *Plan π_i norm-dominates π_j denoted as $(\pi_i, \pi_j) \in \geq_N$*

1. *if $\trianglerighteq_N$ is a total preorder on $N_{vol(\Pi)}$ and $(N_{vol(\pi_i)}, N_{vol(\pi_j)}) \in \trianglerighteq_N$; Else (i.e. if $\trianglerighteq_N$ is not a total preorder on G_Π)*
2. *if $|N_{vol(\pi_i)}| \geq |N_{vol(\pi_j)}|$.*

$>_N$ is the strict relation associated with $\geq_N$. $(\pi_i, \pi_j) \in \sim_N$ iff $(\pi_j, \pi_i) \in \geq_N$ and $(\pi_i, \pi_j) \in \geq_N$.

It is straight forward to see $>_N$ is irreflexive, antisymmetric and transitive, while $\sim_N$ is reflexive, symmetric and transitive.

Goal dominance and norm dominance can be combined in various ways to provide the basis for plan comparison. Generally we are in favour of plans that are goal-dominant, while they are not norm-dominant. In what follows we give priority to dominance of goals over norms. The dominance of norms can be given priority over dominance of goals by swapping the order of conditions 2 and 3 in the following definition.

Definition 41 (Plan Comparison)**.** *Plan π_i is better than π_j, denoted $(\pi_i, \pi_j) \in >_\pi$, iff:*

1. *π_i is justified and π_j is not; or*
2. *π_i and π_j are both justified and $(\pi_i, \pi_j) \in >_G$; or*
3. *π_i and π_j are both justified and $(\pi_i, \pi_j) \in \sim_G$ but $(\pi_j, \pi_i) \in >_N$.*

Plan π_i is as good as π_j, denoted $(\pi_i, \pi_j) \in \sim_\pi$, iff $(\pi_i, \pi_j) \notin >_\pi$ and $(\pi_j, \pi_i) \notin >_\pi$.

Property 5.1. *$>_\pi$ is irreflexive* [6].

Property 5.2. *$>_\pi$ is antisymmetric.*

Property 5.3. *$>_\pi$ is transitive.*

Property 5.4. *$\sim_\pi$ is an equivalence relation on Π.*

Definition 42. *Given $\pi \in \Pi$, let $[\pi_i]$ denote the equivalence class to which π_i belongs. $([\pi_i], [\pi_j]) \in \geq$ iff $(\pi_i, \pi_j) \in >_\pi$ or $(\pi_i, \pi_j) \in \sim_\pi$.*

Property 5.5. *$\geq$ is a total order on Π.*

Definition 43 (Best Plan)**.** *Plan π_i is the best plan for the agent to execute iff*

- *π_i is justified, and*
- *$\nexists \pi_j$ such that $([\pi_j], [\pi_i]) \in \geq$.*

Based on this definition, there might be more than one plan identified as a best plan. Essentially, all members of class $[\pi_i]$ are the best plans. In a multi-agent setting this causes complications, since the agents need additional coordination to agree on a plan out of the best ones, however in a case of a single agent, any of these plan can be chosen at random. On the other hand, there is also the issue of not finding a best plan at all. If none of the agent's plans are justified, the best plan does not exist. Whether the agent is supposed to do nothing at all or just pick a plan randomly, is a subjective view. Since each plan at least satisfies a goal, the agent might as well pick any plan randomly and execute it, but what it executes is not the best plan and is certainly not justifiable. The next section shows a scenario in which the agent uses the criterion in the last definition to identify the best plan.

5.3 Example

In this scenario, we are presenting an agent that has few goals to satisfy, while there are norms imposed on the agent that are triggered based on the actions that it takes to satisfy its goals.

[6]Proof of the properties in this section are provided in Appendix B.

The triggered norms can be complied with or violated, however, the agent has to see the impact of each of these options in a bigger picture (e.g. complying with or violating a norm might prevent satisfying a goal or complying with another norm). We first give an account of the agent's goals and norms imposed on it, and also explain the conflicts between these entities that the agent needs to consider in order to construct conflict-free plans. We then show how the agent identifies the best plan by considering the justifiability of plans, the goals satisfied and norms complied with and violated in each plan.

Let us assume the agent has three goals as presented in Figure 5-6. To satisfy goal $strike$, the agent has to become a union member, not go to the office and not attend any meeting on behalf of the company outside the office. Goal $submission$ is satisfied if the agent goes to the office and finalise a report that is due. Evidently, the agent cannot submit the report while on strike. In addition, the agent also wants to satisfy goal $certificate$, which shows that the agent has undertaken certain training. This goal is satisfied if the agent pays the fee for the training course, takes part in a theory test and attends an interview. If the agent takes the theory test norm n_4 (see Figure 5-7) obliges the agent to attend the interview within two weeks of taking the theory test. But the agent does not have enough saving to pay for the course fee to begin with! The agent, however, can use company's funding to pay the fee for the course she wants to join, but that subject her to norm n_1 (see Figure 5-7). This norm obliges the agent to attend a meeting on behalf of the company within two weeks of receiving such a funding. But then again, if the agent is on strike, she cannot attend any meeting on behalf of the company! In addition to all of these complications, the agent is contemplating going on an extended maternity leave for six weeks. If she goes on maternity leave, she is forbidden to go to the office (norm n_2) or attend any meeting on behalf the company (norm n_3) for the period of 6 weeks. Not going to the office and not attending any meeting although are aligned with satisfying goal $strike$, they risk satisfying goal $submission$ for which the agent has to go to the office and complying with norm n_1, for which the agent has to attend a meeting on behalf of the company.

The actions available to the agent are provided in Figure 5-8. Given the goals and norms discussed above (Figure 5-6 and 5-7), and the following preferences $\succeq^{gn} = \{(submission, n_2), (n_4, n_1)\}$, the agents wants to know what courses of actions are available with respect to conflicts within and between goals and norms and what is the best course of actions to execute. Let us first identify the conflict:

- Conflicting goals: According to Definitions 15 (page 67), we know: $cf_{goal} = \{(strike, submission)\}$ because they require the agent to be and not to be at_office.

$strike = \{union\_member, \neg at\_office, \neg meeting\_attended\}$
$submission = \{at\_office, report\_finalised\}$
$certificate = \{course\_fee\_paid, theory\_test\_done, interviewed\}$

Figure 5-6: Agent Goals

$n1 = \langle o, get\_company\_funding, attend\_meeting, 2\rangle$
$n2 = \langle f, take\_maternity\_leave, go\_to\_office, 6\rangle$
$n3 = \langle f, take\_maternity\_leave, attend\_meeting, 6\rangle$
$n4 = \langle o, take\_theory\_test, attend\_interview, 2\rangle$

Figure 5-7: Agent Norms

- Conflicting goals and norms: According to Definition 18 (page 69) $cf_{goalnorm} = \{(strike, n_1), (submission, n_2)\}$.
  - $(strike, n_1)$: norm n_1 obliges the agent to attend a meeting on behalf of the company, whereas if the agent want to go on strike, it should not attend any meeting.
  - $(submission, n_2)$: satisfying goal $submission$ requires the agent to go to office, however this is forbidden by norm n_2.
- Conflicting norms: According to Definition 19 due to the concurrency conflict between $attend\_meeting$ and $attend\_interview$, and depending on the way actions are sequenced, it is possible that in some plan π, $(n_1, n_4) \in cf^{\pi}_{norm}$. Also according to Definitions 20, it is possible that in some plan π, $(n_1, n_3) \in cf^{\pi}_{norm}$, because n_1 obliges the agent to execute $attend\_meeting$, while n_3 prohibits the agent from doing so.

In what follows we look at four plans[7] for the agent: $\pi_1, \pi_2, \pi_3, \pi_4 \in \Pi$. In the previous section (page 120) we provided an example of identifying goal- and norm-dominance according to preferences. In this example we highlight identifying goal- and norm-dominance when the preference information is not sufficient to recognise these features. Instead, goal- and norm-dominance are identified based on the number of goals satisfied and norms violated in each plan.

$$\pi_1 = \langle (get\_company\_funding, 0), (go\_to\_meeting\_venue, 1), (attend\_meeting, 2), \\ (go\_to\_office, 7), (finish\_report, 8)\rangle$$

[7] These plans are generated by the execution of the ASP code of this example in Appendix A.

$$join\_union = \langle\{\neg union\_member\}, \{union\_member\}, 1\rangle$$

$$finish\_report = \langle\{at\_office\}, \{report\_finalised\}, 3\rangle$$

$$go\_to\_office = \langle\{\neg at\_office\}, \{at\_office\}, 1\rangle$$

$$attend\_interview = \langle\{theory\_test\_done, at\_interview\_venue, \neg at\_office, \neg at\_meeting\_venue\}, \{interviewed, \neg at\_interview\_venue\}, 4\rangle.$$

$$take\_theory\_test = \langle\{course\_fee\_paid\}, \{theory\_test\_done\}, 1\rangle$$

$$get\_company\_funding = \langle\{employee\}, \{course\_fee\_paid\}, 2\rangle$$

$$attend\_meeting = \langle\{course\_fee\_paid, at\_meeting\_venue, \neg at\_office, \neg at\_interview\_venue\}, \{meeting\_attended, \neg at\_meeting\_venue\}, 5\rangle.$$

$$go\_to\_interview\_venue = \langle\{\neg at\_interview\_venue\}, \{at\_interview\_venue, \neg at\_office, \neg at\_meeting\_venue\}, 1\rangle$$

$$go\_to\_meeting\_venue = \langle\{\neg at\_meeting\_venue\}, \{at\_meeting\_venue, \neg at\_office, \neg at\_interview\_venue\}, 1\rangle$$

$$take\_maternity\_leave = \langle\{given\_birth\}, \{on\_maternity\_leave\}, 2\rangle$$

Figure 5-8: Agent Actions

$$\pi_2 = \langle (go\_to\_office, 0), (get\_company\_funding, 1), (finish\_report, 2),$$
$$(take\_theory\_test, 4), (go\_to\_interview\_venue, 5), (attend\_interview, 6)\rangle$$

$$\pi_3 = \langle take\_maternity\_leave, 0), (get\_company\_funding, 1),$$
$$(go\_to\_office, 2), (finish\_report, 3), (take\_theory\_test, 5),$$
$$(go\_to\_interview\_venue, 6), (attend\_interview, 7)\rangle$$

$$\pi_4 = \langle (get\_company\_funding, 0), (take\_maternity\_leave, 1),$$
$$(take\_theory\_test, 2), (go\_to\_interview\_venue, 3),$$
$$(attend\_interview, 4), (join\_union, 5)\rangle$$

Table 5.2 shows the goals satisfied in each plan as well as norms complied with and violated. In order to identify the best plan, we first need to identify the justified plans which were defined as credulously accepted under the preferred semantics in Definition 35. Figures 5-9, 5-10, 5-11 and 5-12 display the DAF [8] associated with each plan. Note that the set of activated norms differs from one plan to another plan, hence different set of arguments in each graph. For example norm n_4 was never activated in plan π_1, because action $take\_theory\_test$ was not executed in this plan. Therefore, Arg_{n_4} does not exist in Figure 5-9, whereas, it does exist in Figures 5-10, 5-11 and 5-12. Also note how Arg_{n_4} attacks Arg_{n_1} in the context of plan π_2 and π_4, while that is not the case in the context of plan π_3. This shows how the conflict between norms is contextual and differs from one plan to another one.

Having computed the preferred extensions [9] of DAF_{π_1}, DAF_{π_2}, DAF_{π_3}, and DAF_{π_4}, it turns out that Arg_{π_1} is not credulously accepted, whereas Arg_{π_2}, $Arg_{\pi_3}$and Arg_{π_4} all are. Thus, plan π_1 is not justified, whereas π_2, π_3 and π_4 are. Based on Definition 41 we have: $(\pi_2, \pi_1), (\pi_3, \pi_1), (\pi_4, \pi_1) \in >$. Also, since $(\pi_2, \pi_3) \in \sim_G$, while $(\pi_3, \pi_2) \in >_N$, it follows that $(\pi_2, \pi_3) \in >$. Moreover, $(\pi_2, \pi_4) \in \sim_G$ and $(\pi_2, \pi_4) \in \sim_N$, which makes them $(\pi_2, \pi_4) \in \sim$. According to Definition 43, plan π_2 and π_4 are both identified as the best plans for the agent to execute.

[8] st, sub and cer in these figures stand for $strike$, $submission$ and $certificate$ respectively.

[9] Online argumentation tool such as ASPARTIX (http://rull.dbai.tuwien.ac.at:8080/ASPARTIX/index.faces) can compute the preferred extensions of these graphs in less than 5 seconds.

	G_{π_i}	$N_{cmp(\pi_i)}$	$N_{vol(\pi_i)}$
π_1	$\{submission\}$	$\{n_1\}$	$\{\}$
π_2	$\{submission, certificate\}$	$\{n_4\}$	$\{n_1\}$
π_3	$\{submission, certificate\}$	$\{n_3, n_4\}$	$\{n_1, n_2\}$
π_4	$\{certificate, strike\}$	$\{n_2, n_3, n_4\}$	$\{n_1\}$

Table 5.2: Goals and Norms Satisfied, Complied with and Violated

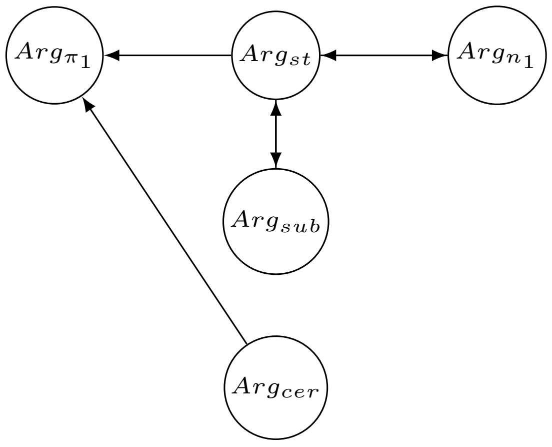


Figure 5-9: Argumentation Framework for plan π_1

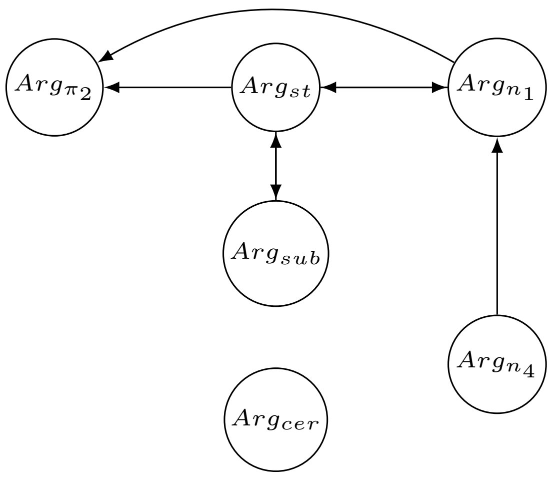


Figure 5-10: Argumentation Framework for plan π_2

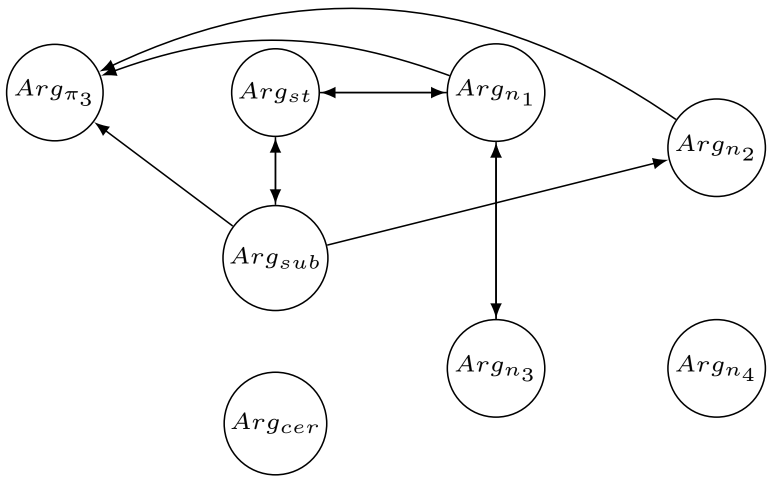


Figure 5-11: Argumentation Framework for plan π_3

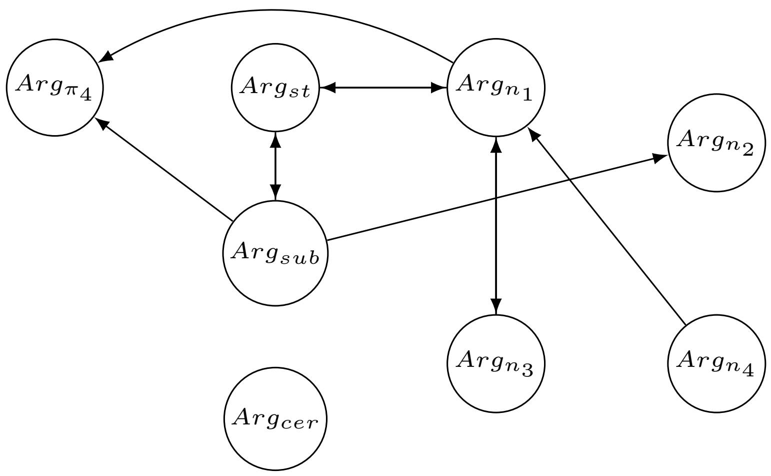


Figure 5-12: Argumentation Framework for plan π_4

5.4 Discussion

In this section, we revisit and discuss four choices made in this chapter with regards to the structure of arguments, the use of preferences, the justifiability of plans, and the decision criterion to compare plans.

Argument structure: We use argument schemes to represent the structure of arguments as opposed to structure-based approaches such as ABA [Dung et al., 2009] and ASPIC$^+$ [Caminada and Amgoud, 2007]. We explain the rationale behind this choice in Section 5.1.1. Very recently a new version of ASPIC$^+$, called ASPIC^+_D, with purely defeasible rules is proposed by Li and Parsons [2015]. A mapping of our framework to ASPIC^+_D, where arguments are represented using the defeasible structured-based approach, should be easily obtainable. Although, it is interesting from a representation perspective, this mapping would not affect the output of the framework. In fact using ASPIC^+_D, any scheme-based argumentation framework should be mappable to a structure-based one.

Preferences: The role of preferences in distinguishing between attacks and defeats is not subjective. It is always the case the attack from an argument to another one is identified as a defeat if the latter argument is not preferred over the former. However, the role of preferences in refinement of the evaluation of arguments is a subjective matter. We used the democratic principle to further compare justified plans due to its effectiveness in dealing with argumentation framework with even cycles of arguments. However, depending on the properties of the argumentation framework a variety of set ordering techniques can be used to refine the evaluation of arguments. A survey of methods to lift the preferences between objects to preferences over sets that include those objects can be found [Barber et al., 2001]. Using different principles, many different decision criterions for comparing plans can be created that deal with different situations (e.g. when a set has the least favourable object in comparison with another set, or when a set has the most favourable object in comparison with another set). What we wanted to show in comparison of plans was how the quality (i.e. preferences) and quantity of objects in sets (i.e. goals and norms satisfied, violated and complied with in different plans) can be combined to provide a ranking over the agent's plans.

Another point worth mentioning is that preferences in our work are expressed between individual goals and norms. Fan et al. [2013] use partial preferences defined over individual goals and also sets of goals for decision-making with preferences in a medical domain. Their notion of preferences over sets of goals can, in our work, be extended to preferences over a combination of goal and norm sets, which will be used to identify the best plan out of the justified set of plans. However, such an extension requires using a Dung-style argumentation framework instead of preference-based. We show the importance of this point in the following example.

Assume that the agent has three goals g_1, g_2, g_3 with the following preferences $(\{g_2, g_3\} \succ \{g_1\} \succ \{g_2\} \succ \{g_3\})$. There are two plans, π_1 and π_2, available to agent, such that the former plan satisfies g_1 and the latter satisfies g_2 and g_3. Goal g_1 is in conflict with both g_2 and g_3, which means that $(Arg_{g_1}, Arg_{g_2}), (Arg_{g_2}, Arg_{g_1}), (Arg_{g_1}, Arg_{g_3}), (Arg_{g_3}, Arg_{g_1}) \in Att$. Since $g_1 \succ g_2$ and $g_1 \succ g_3$ the following attacks are identified as defeats $(Arg_{g_1}, Arg_{g_2}), (Arg_{g_1}, Arg_{g_3}) \in Def$ which makes Arg_{g_1} the only justified argument. Consequently, plan π_1 that satisfies g_1 is justified whereas, plan π_2 that satisfies g_2 and g_3 is not. By definition all justified plans are better than non-justified ones, which means that π_1 is better than π_2. Thus the following preference $\{g_2, g_3\} \succ \{g_1\}$ is never used.

Now let us eliminate using preferences to distinguish between attacks and defeats. We get two sets of justified arguments: $\{Arg_{g_1}\}$ and $\{Arg_{g_2}, Arg_{g_3}\}$ and because $\{g_2, g_3\} \succ \{g_1\}$ plan π_2 is better than π_1.

Justifiability: We check the justifiability of plans in a distributed manner. There is an argumentation framework associated with each plan proposal that gets evaluated based on the preferred semantics and consequently results in identifying the justified plans. Since the set of activated norms differs from one plan to another one, the set of norm arguments also differs from one framework to another one. In addition, the attack between norm arguments varies from one framework to another one. It is therefore, impossible to check the justifiability of plans all in the same framework.

Multi-criteria decision-making: In the decision criterion defined in Definition 41, we compare the justified plans based on the set of goals they satisfy and if they are equally good from that perspective, we look at the set of norms that they violate. The order can easily be reversed, meaning that justified plans can first be compared based on the set of norms they violate and if they are equal from that viewpoint, they can then be compared based on the set of goals they satisfy. The method of comparing plans can be seen as a type of argumentation-based multi-criteria decision-making. In multi-criteria decision-making theory, a set of candidate decisions are evaluated from the perspective of a set of criteria. In the argumentation-based version of this theory [Amgoud and Prade, 2009], arguments are built in favour and against a decision from each criteria perspective. The satisfaction and dissatisfaction of a criteria count as pros and cons respectively and ultimately the decisions are ranked depending on the strength or number of pros and cons and the importance of the criteria. In our case, it could be assumed that the plans are the decisions and the agent goals and norms are the criteria to compare the decisions. Satisfying goals counts as a pro and violating a norm counts as a con. Depending on the order on considering goal- or norm-dominance, a plan that has the most (in number of strength) pros is better than another plan with fewer pros. If plans cannot be compared based on their pros, they will be compared based on their cons, such that a plan with the fewer (in number of strength) cons is a better plan.

5.5 Summary

In this chapter we built on recent work in argumentation theory to reason about justifiability of plans as a step toward identifying the best plan. The arguments are structured based on argumentation schemes for plans, goals and norms. The critical questions associated with these schemes provide the ways in which arguments can attack each other. The arguments and their relationships constitute the main elements of an argumentation framework for plan proposals. Furthermore, we discussed the need for taking the agent preferences into account in the constructed argumentation framework, hence the use of preference-based argumentation framework. We also made the case for the use of the preferred semantics to evaluate this framework, with the aim of identifying the justified plans. Justified plans were further compared based on the fact that a plan that satisfies more important goals or more goals, while it violates fewer important norms or fewer norms, in comparison with another plan, is the better plan of the two.

In Chapter 3 we defined the agent's plans, followed by generating them in Chapter 4. This chapter focused on identifying the best plan using argumentation. In the next chapter we discuss the processes for generating an explanation in natural language for the best plan.

Chapter 6

Explaining The Best Plan via Dialogue

In the previous chapter we proposed an argumentation-based approach for identifying the best plan. More precisely, we used the preferred semantics to examine the justifiability of a plan and used the goals satisfied and norms violated in each plan to compare justified plans in search of the best plan. In essence, Chapter 5 provided an answer for *how should an agent act in a normative environment when it has conflicting goals and norms?* Existing frameworks, other than the argumentation-based ones, that deal with this question [Broersen et al., 2001; Criado et al., 2010; Kollingbaum and Norman, 2003] have not paid any attention to explaining the agent's reasoning and decision-making. It is therefore, quite challenging for humans to understand and ultimately trust these frameworks and their outcomes. To address this shortcoming, in the previous chapter, we promoted the use of argumentation as a reasoning tool, while in this chapter we use the dialogical aspect of argumentation to generate explanation for the reasoning and its outcomes. Thus, in this chapter we deal with the question of *how can the agent explain in natural language why it acted in a certain way?*

Assuming that the agent acted on the best plan, in order to answer why it acted that way, the agent has to explain why the executed plan is the best plan. Explaining the best plan involves generating an explanation for why the plan is justified and why none of the other justified plans are a better option. The justifiability of a plan can be explained by simply acknowledging the membership of the argument for that plan in a specific set (i.e. a preferred extension). But such an acknowledgement on its own does not provide a concrete explanation that is understandable to a human. Although many scholars [Amgoud, 2012; Kakas and Moraitis, 2003; Oren, 2013; Rahwan and Amgoud, 2006] promote argumentation-based approaches to practical reasoning and decision-making due to its explanatory power and its alignment with human decision-making, they have rarely used argumentation to generate explanation that is accessible to human users. That is with the exception of two recent works by Zhong et al. [2014] and

Caminada et al. [2014c], where the authors use argumentation-based dialogues coupled with natural language generation to provide an explanation that is accessible for non-expert users as well as expert ones. In the former the authors translate the admissible dispute trees developed for assumption-based argumentation [Dung et al., 2009] to natural language for decision-making in a legal scenario. In Caminada et al. [2014c], a dialogical proof procedure based on the grounded semantics dialogue game [Caminada and Podlaszewski, 2012b] is created to justify the actions executed in a plan. The justification is mainly focused on the preconditions and effects of actions with regards to the goal state. The reasons behind the choice of the grounded semantics are not clear.

In this chapter, we use argumentation-based dialogue for the preferred semantics translated into natural language to explain the justifiability of the best plan. The contribution of this chapter is in applying a newly developed dialogue game, called *Socratic Discussion* [Caminada et al., 2014b], to a normative practical reasoning problem, where the agent uses the dialogue as an internal dialogue to bring transparency to the procedure of its decision-making about what plans are justified with respect to the agent's goals, norms, their conflicts and preferences. To the best of our knowledge this work is the first application of standard dialogue games for the preferred semantics in general and Socratic discussion in particular.

In order to provide a full explanation for the best plan, in addition to explaining the justifiability of the plan, the agent has to explain why none of the other justified plans are a better option to execute. We use simple algorithms in the spirit of Zhong et al. [2014] to explain such a comparison. The algorithm in Zhong et al. [2014], however, is specific to a particular legal domain, and it checks the justifiability under admissible semantics in assumption-based argumentation. Therefore, we develop algorithms based on the preferred semantics and the definitions of goal- and norm-dominance that can be used by any normative practical reasoning agent to explain its best plan.

This chapter is organised as follows. In the next section we give a brief survey of existing dialogue games for the preferred semantics. Section 6.1.1 focuses on a particular dialogue game for the preferred semantics known as Socratic discussion. This discussion will be used in Section 6.1.2 to explain the justifiability of the best plan. The full explanation of the best plan in natural language is provided in Section 6.2 followed by a discussion section in 6.3. The summary of the chapter can be found in Section 6.4.

6.1 Dialogue Game for the Preferred Semantics

In Chapter 2 (page 45) we explained that argumentation semantics specify what arguments are justified in a framework, without indicating the procedure for construction of such a set of justified arguments [Fan and Toni, 2015; Jakobovits and Vermeir, 1999]. Therefore, many (e.g. [Caminada, 2004; Caminada and Podlaszewski, 2012b; Jakobovits and Vermeir, 1999; Modgil and Caminada, 2009; Prakken, 2006a]) believe that the study of semantics in terms of dialogues can decrease the gap between intuitive and formal accounts of argumentation. Such dialogue games also known as argument games, dialectical proof theories or dispute trees aim at testing membership of an argument in an extension under certain semantics [Jakobovits and Vermeir, 1999; Prakken and Sartor, 1997; Vreeswijk and Prakken, 2000].

The question of whether an argument is credulously accepted under the preferred semantics[1] (i.e. it is a member of at least one preferred extension) has been addressed few times after the preferred semantics were introduced by Dung [1995]. That is despite the pessimistic results on computational complexity of proof theories for the preferred semantics, that were presented in Dimopoulos and Torres [1996]. Vreeswijk and Prakken [2000], however argue why it is nevertheless important to motivate a proof theory for the preferred semantics. A detailed account of their reasons can be found in Vreeswijk and Prakken [2000]. Their main arguments are (i) the pessimistic results only concern worst-case scenarios and it is not the case all the time; and (ii) the dialogues corresponding to logics for defeasible argumentation can be interrupted at any time while the intermediate outcome is still meaningful (sic). In addition, the obtained results in Dimopoulos and Torres [1996] reports that the credulous reasoning under the admissible and preferred and stable semantics has the same complexity. However, sceptical reasoning under the preferred semantics has a higher complexity than under the stable and admissible semantics. Thus, the pessimistic results solely concerns the sceptical reasoning under the preferred semantics.

In the remainder of this section we briefly survey the developed dialogues for the preferred semantics. All these dialogue games use the same line of reasoning: by definition (page 34 Chapter 2) a preferred extension is a maximal admissible set with respect to set inclusion. Therefore, an argument that is in at least one preferred extension is in at least one admissible extension. As a result, deciding if an argument is in at least one preferred extension amounts to deciding if it is at least in one admissible extension.

Vreeswijk and Prakken [2000] presented one of the first dialogue games for the preferred semantics. Two parties in this dialogue are PRO and CON, standing for proponent and opponent. A *move* is an argument that is put forward first in the dialogue, or an argument attacking the previous argument of the previous player. Both players are allowed to *backtrack*, however an *eo ipso* is a move using a previous non-backtracked argument of the other player. A *block* is another type of move that puts the other player in a position, where the player cannot

[1]In the rest of this chapter, unless it is specified otherwise, for simplicity, we use the preferred semantics to refer to the credulous type of the preferred semantics.

make any move. A TPI-dispute (two-party immediate response dispute) is a dispute in which both players are allowed to repeat PRO; PRO is not allowed to repeat CON; and CON is allowed to repeat CON. The winner of the dialogue is CON, if it uses move *eo ipso* or *blocks* PRO. Otherwise, PRO is the winner of the dialogue. The initial argument in the dialogue is defended if PRO wins the dispute. It is proved that an argument is accepted under the preferred semantics iff it can be defended in every TPI-dispute.

Another proof theory for the preferred semantics was proposed by Cayrol et al. [2001]. It is claimed that this proof theory improves on Vreeswijk and Prakken [2000] by providing a shorter proof for a given argument. The reason for the shortness is that Cayrol et al. [2001] prevent proponent (PRO) and opponent (OPP) from moving with arguments that are the successors of arguments advanced by PRO during the dialogue. These arguments are forbidden for PRO, because we want the set of arguments advanced by PRO to remain conflict-free. They are also forbidden by OPP because they are rejected from any admissible set containing arguments advanced by PRO, so there is no point for OPP to advance these arguments.

Modgil and Caminada [2009] specify argument games for the preferred semantics based on Caminada's labellings [Caminada, 2006]. Similar to the above games, these argument games test if an argument put forward by a proponent is labelled in by at least one admissible labelling. Admissible labellings are formally defined in the next section.

Caminada et al. [2014b], and Caminada [2010] believe that the dialogue games for the preferred semantics are merely technical and mathematical, making it difficult to grasp the reasoning concept behind the semantics. In order to bridge the gap between the mathematical formulation and the philosophical intuition behind the semantics, they make a connection between the preferred semantics and Socratic discussion. In Socrates's form of reasoning, as documented by Plato, the first party (referred to as proponent and denoted as M that stands for Menexenus) puts forward a claim and tries to defend it by avoiding being lead to a contradiction by a second party (referred to as opponent and denoted as S that stands for Socrates). The proponent committing itself to a contradiction shows that its arguments cannot hold together, which ultimately shows the absurdity of its original claim. The connection made between the preferred semantics and Socratic discussion is formulated in the form of a dialogue game that like other dialogues checks the acceptability of an argument under the preferred semantics by checking the membership of the argument in at least one admissible extension. Since the formulation of the preferred semantics in terms of Socratic discussion is the most intuitive available dialogue game for this semantics, we use this dialogue to explain the justifiability of best plan. In the next section we give a formal account of Socratic discussion.

6.1.1 Preferred Semantics as Socratic Discussion

As established in the previous section, deciding if an argument is in at least one preferred extension amounts to deciding if it is at least in one admissible extension. Translating this statement in terms of Caminada's labellings, it follows that an argument is in at least one admissible set iff it is labelled in by at least one admissible labelling [Caminada and Gabbay, 2009]. Consequently, an argument is in at least one preferred extension if it is labelled in by at least one admissible labelling. Definition 44 describes admissible labellings.

Definition 44 (Admissible Labellings [Caminada and Gabbay, 2009]). *Let $\langle Arg, Def\rangle$ be an argumentation framework, a labelling is an admissible labelling iff for every $a \in Arg$ it holds that: if a is labelled in then all attackers of a are labelled out; and if a is labelled out then a has at least one attacker that is labelled in.*

Socratic discussion is a discussion based on admissible labellings. If player M puts forward an argument a as labelled in and wins the discussion, then a is in at least one admissible extension. Since an argument is in at least one admissible extension iff it is in at least one preferred extension, we conclude that if play M wins the Socratic discussion, the original argument put forward is accepted under the preferred semantics.

Definition 45 (Socratic Discussion [Caminada et al., 2014b]). *Let $AF = \langle Arg, Def\rangle$ be an argumentation framework. The sequence of moves $[\Delta_1, \Delta_2, \cdots, \Delta_n]$ $(n \geq 1)$ is a Socratic discussion iff:*

- *each odd move (called M-move): Δ_i $(1 \leq i \leq n, i \in 2\mathbb{Z}^+ + 1)$ is an argument labelled in: $in(a), a \in Arg$;*
- *each even move (called S-move): Δ_i $(2 \leq i \leq n, i \in 2\mathbb{Z}^+)$ is an argument labelled out: $out(a), a \in Arg$;*
- *each argument moved by S attacks by an argument moved by M: $\forall\Delta_i = out(a), (2 \leq i \leq n, i \in 2\mathbb{Z}^+), \exists\Delta_j = in(b), (j < i, j \in 2\mathbb{Z}^+ + 1)$ s.t. $(a, b) \in Def$;*
- *each argument moved by M attacks an argument moved by S in the previous step: $\forall\Delta_i = in(a), (3 \leq i \leq n, i \in 2\mathbb{Z}^+ + 1), \exists\Delta_{i-1} = out(b)$ s.t. $(a, b) \in Def$; and*
- *S-moves cannot be repeated: $\nexists\Delta_i, \Delta_j$ s.t. $i \neq j$ and $\Delta_i = \Delta_j$*

There are two conditions for discussion $[\Delta_1, \Delta_2, \cdots, \Delta_n]$ to be considered as finished (i) there exists no move Δ_{n+1} such that $[\Delta_1, \Delta_2, \cdots, \Delta_n, \Delta_{n+1}]$ is a discussion, or there exists an M-move and an S-move containing the same argument; and (ii) no subsequence $[\Delta_1, \Delta_2, \cdots, \Delta_m]$ $(m < n)$ is finished. Player S wins the discussion if there exists an M-move and an S-move containing the same argument. Otherwise, the winner of the discussion is the player that makes the last move.

Figure 6-1 shows examples of Socratic discussion that are won by the opponent (S) (top figure) and the proponent M (bottom figure), respectively. In the next section we show how we use this discussion to check the membership of arguments for plan proposals in a preferred extension.

6.1.2 Socratic Discussion for Explaining the Justified Plans

From the definition of the best plan in the previous chapter (page 122) we know that the best plan is justified, meaning that it is accepted under the preferred semantics. As established in the previous section, since the best plan (e.g. π) is accepted under the preferred semantics, given DAF_π, player M is guaranteed at least one winning strategy in a Socratic discussion with $\Delta_1 = in(Arg_\pi)$. According to Property 5.1 (Chapter 5, page 115) Arg_π is labelled in iff all the arguments for the goals that it does not satisfy and the norms that it violates are labelled out. Therefore, the winning strategy for Arg_π is composed of the agent justifying why it did not satisfy a goal or why it violated a norm. In doing so, the agent is dialectically pointing out the reasons for not satisfying a goal or violating a norm, while the reasons are satisfying or complying with a more preferred goal or norm. It could also happen that there were no preferences expressed over a goal or a norm, in which case the agent expresses the reason for satisfying one as opposed to another one by stating that due to their conflict only one of them had to be satisfied or complied with.

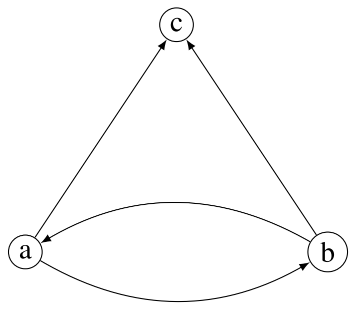
Let $DAF_\pi = \langle Arg, Def \rangle$ be an argumentation framework for plan π that happens to be the best plan. Also let the sequence of moves $\Delta = [\Delta_1, \Delta_2, \cdots, \Delta_n]$ $(n \geq 1)$ be a Socratic discussion for plan π. Algorithm 1 provides a translation of the Socratic discussion for this plan in natural language. Note that this algorithm uses Property 5.1 and 5.2 (Chapter 5, page 115) to translate (i) the arguments that are labelled in as the goals and norms satisfied and complied with in the plan; (ii) the arguments that are labelled out as the goals and norms not satisfied and violated in the plan.

6.2 Explaining the Best Plan

In this section we show a simple algorithm (Algorithm 2) that explains why plan π is the plan for the agent to execute. The algorithm takes the following as input: the best plan π, the set of justified plans Π_{Just}, the set of satisfied goals in each plan $G_\Pi = [G_{\pi_1}, G_{\pi_2}, \cdots, G_{\pi_n}]$, the set of violated norms in each plan $N_{vol(\Pi)} = [N_{vol(\pi_1)}, N_{vol(\pi_2)}, \cdots, N_{vol(\pi_n)}]$ and set of preferences over goals and norms $\succ^{gn}$. The output of the algorithm is (i) the explanation of the justifiability of the best plan using the $Justified$ function in Algorithm 1; and (ii) the

Algorithm 1: Explanation the Justifiability of the Best Plan in Natural Language

```
function: Justified($\pi$, $DAF_\pi$, $\Delta$)
output  : Plan $\pi$ is justified.
for all $\Delta_i = out(Arg_x)$ do
    if $Arg_x \in Arg_G$ then
        output  : Why the plan does not satisfy goal $x$?
        Let $\Delta_{i+1} = in(Arg_y)$
        if $Arg_y \in Arg_G$ then
            if $(Arg_x, Arg_y) \in Def$ then
                output  : because goal $y$ and goal $x$ are in conflict and there is no
                          preference expressed between them, so the plan satisfies goal
                          $y$ instead of $x$.
            else
                output  : because goal $y$ and goal $x$ are in conflict and goal $y$ is
                          preferred to goal $x$, so the plan satisfies goal $y$ instead of $x$.
        else
            if $(Arg_x, Arg_y) \in Def$ then
                output  : because norm $y$ and goal $x$ are in conflict and there is no
                          preference expressed between them, so the plan complies with
                          norm $y$ instead of satisfying goal $x$.
            else
                output  : because norm $y$ and goal $x$ are in conflict and goal $x$ is
                          preferred to norm $y$, so the plan complies with norm $y$ instead
                          of satisfying goal $x$.
    else
        output  : Why the plan violates norm $x$?
        Let $\Delta_{i+1} = in(Arg_y)$
        if $Arg_y \in Arg_G$ then
            if $(Arg_x, Arg_y) \in Def$ then
                output  : because goal $y$ and norm $x$ are in conflict and there is no
                          preference expressed between them, so the plan satisfies goal
                          $y$ instead of complying with norm $x$.
            else
                output  : because goal $y$ and norm $x$ are in conflict and goal $y$ is
                          preferred to norm $x$, so the plan satisfies goal $y$ instead of
                          complying with norm $x$.
        else
            if $(Arg_x, Arg_y) \in Def$ then
                output  : because norm $y$ and norm $x$ are in conflict and there is no
                          preference expressed between them, so the plan complies with
                          norm $y$ instead of complying with norm $x$.
            else
                output  : because norm $y$ and norm $x$ are in conflict and norm $y$ is
                          preferred to norm $x$, so the plan complies with norm $y$ instead
                          of complying with norm $x$.
```

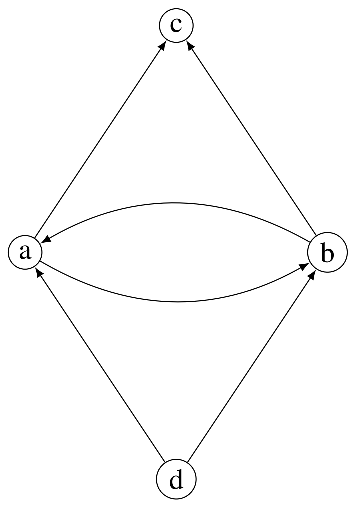


M: in(c)
"Argument c is justified and therefore labelled in."

S: out(a)
"If you claim c is justified then its attacker a must be labelled out.

M: in(b)
"a is labelled out because b is labelled in and b attacks a."

S: out(b)
"But from your original claim that c is labelled in, it holds that b that is the attacker of c should be labelled out. However that contradicts the claim that you just made that b is labelled in. Therefore your original claim that c is rejected."



M: in(c)
"Argument c is justified and therefore labelled in."

S: out(a)
"If you claim c is justified then its attacker a must be labelled out. Why is that? "

M: in(d)
"a is labelled out because d is labelled in and d attacks a."

S: out(b)
"But b also attacks c, So if you claim c is justified then its attacker b must be labelled out too."

M: in(d)
"b is labelled out because d is labelled in and d attacks a."

Figure 6-1: Example of Socratic discussion

Algorithm 2: Explanation the Best Plan in Natural Language

```
function : Bestplan($\pi, \Pi_{Just}, G_{\Pi}, N_{vol(\Pi)}, \succeq^{gn}$)
Justified($\pi$, $DAF_{\pi}$, $\Delta$)
for all $\pi' \in \Pi_{Just} \setminus \pi$ do
    output : Why plan $\pi'$ is not better than plan $\pi$?
if $(\pi, \pi') \in >_G$ then
    output : Because plan $\pi$ goal-dominates plan $\pi'$
    if $(G_{\pi}, G_{\pi'}) \in \trianglerighteq_G$ then
        output : since for every goal satisfied in $\pi'$ that is not satisfied in $\pi$, there is at least
                 one preferred goal satisfied in $\pi$ that is not satisfied in $\pi'$.
    else
        output : since the number of goals satisfied in plan $\pi$ is more than the number of
                 goals satisfied in plan $\pi'$.
else
    if $(\pi, \pi') \in \sim_G$ and $(\pi', \pi) \in >_N$ then
        output : Because plan $\pi$ and $\pi'$ satisfy the same set/number of goals but $\pi'$
                 norm-dominates plan $\pi$
        if $(N_{vol(\pi')}, N_{vol(\pi)}) \in \trianglerighteq_N$ then
            output : since for every norm violated in plan $\pi$ that is not violated in plan $\pi'$,
                     there is at least one preferred norm violated in plan $\pi'$ that is not
                     violated in $\pi$.
        else
            output : since the number of norms violated in plan $\pi'$ is more than the number
                     of norms violated in plan $\pi$.
    else
        output : Because plan $\pi$ and $\pi'$ are equally good, since they satisfy the same
                 set/number of goals and violate the same set/number of norms.
```

explanation of why this plan is better than other justified plans using the definitions of goal-dominance (Definition 38, Page 120), norm-dominance (Definition 40, Page 121) and plan comparison (Definition 41, Page 122), in the previous chapter.

To give an example, we recall the scenario discussed in the previous chapter (page 122). The agent in the scenario has four plans $\pi_1, \pi_2, \pi_3, \pi_4$. Plan π_1 is not justified, whereas π_2, π_3 and π_4 are. Based on Definition 41 we have: $(\pi_2, \pi_1), (\pi_3, \pi_1), (\pi_4, \pi_1) \in >$. Also, since $(\pi_2, \pi_3) \in \sim_G$, while $(\pi_3, \pi_2) \in >_N$, it follows that $(\pi_2, \pi_3) \in >$. Moreover, $(\pi_2, \pi_4) \in \sim_G$ and $(\pi_2, \pi_4) \in \sim_N$, which makes them $(\pi_2, \pi_4) \in \sim$. According to Definition 43, plan π_2 and π_4 are both identified as the best plans for the agent to execute. In figure 6-2, we provide the explanation of plan π_4 that is identified as one of the best, as an example of how a best plan is explained in natural language using Algorithm 1 and 2. Since plan π_2 is also one of the best plans, the same explanation can be provided for plan π_2.

- Plan π_4 is justified.
- Why the plan does not satisfy goal $submission$?
- Because goal $submission$ and goal $strike$ are in conflict and there is no preference expressed between them, so the plan satisfies goal $strike$ instead of goal $submission$.
- Why does the plan violate norm n_1?
- Because norm n_1 and norm n_4 are in conflict in this plan and norm n_4 is preferred to norm n_1, so the plan complies with n_4, instead of complying with norm n_1.[2]
- Why plan π_2 is not better than plan π_4?
- Because plan π_2 and π_4 are equally good, since they satisfy the same number of goals and violate the same set of norms.
- Why plan π_3 is not better than plan π_4?
- Because plan π_3 and π_4 satisfy the same set of goals but plan π_3 norm-dominates π_4, since the number of norms violated in π_3 is more than the number of norms violated in plan π_4.

Figure 6-2: Explanation of plan π_4 in Natural Language

6.3 Discussion

In Section 6.1 we surveyed several dialogues for the preferred semantics. Despite the importance of the preferred semantics and the development of different dialogues for these semantics [Caminada et al., 2014b; Cayrol et al., 2001; Modgil and Caminada, 2009; Vreeswijk and Prakken, 2000], there is not yet any application that uses any of them. In structured argumentation formalism (e.g. ASPIC-style [Caminada and Amgoud, 2007; Modgil and Prakken, 2013]) that is mainly due to the fact that the preferred semantics require the concept of *restricted rebut* to yield consistent outcome. In these formalisms arguments are a chain of defeasible and strict rules. *Restricted* and *unrestricted* rebut [Caminada and Amgoud, 2007] differ in that with restricted rebut, only a conclusion that is the consequent of a defeasible rule can be rebut-attacked, whereas with unrestricted rebut, any conclusion that has at least one defeasible rule involved in its derivation can be rebut-attacked. As a result when using restricted rebut two arguments with contradictory conclusions do not necessarily attack one another, which is not intuitive at all. For example, if argument A and B are as follows[3]: $A = (a \Rightarrow b) \rightarrow c$, $B = (d \rightarrow e) \Rightarrow \neg c$, argument A restrictedly rebut-attacks B, but B does not restrictedly rebut-attack A, however they both attack each other unrestrictedly. Following unrestricted rebut can yield consistent result only under the grounded semantics. For other semantics includ-

[2] Another potential explanation for why the plan violates n_1, is because norm n_1 and norm n_3 are in conflict in this plan and there is no preference expressed between them, so the plan complies with norm n_3 instead of n_1.

[3] $\Rightarrow$ and $\rightarrow$ denote defeasible and strict inference, respectively.

ing the preferred, the application of restricted rebut is required to obtain consistent outcome [Caminada and Amgoud, 2007]. However, the unintuitive nature of restricted rebut in the preferred semantics goes against the motivation behind using dialogue games for this semantics in an agent-to-agent or human-to-agent setting.

Given that we use argumentation schemes to express the internal structure of arguments (Chapter 5, page 107) and these schemes are defeasible by nature, restricted rebut and unrestricted rebut coincide and there is no need to define unrestricted rebut. Hence all the developed dialogues for the preferred semantics, without loss of intuitiveness, are available to the agent to use as a mechanism to explain why a plan is justified.

Another point of discussion is the evaluation pathways for the explanation that is presented to the user. Depending on the level of abstractions different hypothesis can be empirically evaluated by the users of the system. At the very basic level would be to verify that such an explanation is indeed useful. This hypothesis has been evaluated in the context of explanation in expert systems [Moore and Swartout, 1988; Teach and Shortliffe, 1981; Wick, 1992; Ye, 1995], where the results has verified the usefulness of explanation facilities. In terms of the current system this hypothesis can be formally formulated as: "H1: Explanation makes an agent's decisions on course of actions more acceptable to the user."

In a different level, the type of explanation provided can be empirically evaluated. Moulin et al. [2002] categorise the explanation generated in the knowledge-based systems into two broad types. In the firs type, referred to as cooperative or proactive, the system's outcome is always accompanied with an explanation. Whereas, in the second type, referred to as interactive or reactive, the system provides explanation to user's questions rather than generating an overall explanation of the outcome automatically. The first type puts the burden on the system to take into account all the questions that a user might raise and to incorporate a comprehensive answer to the questions in the generated explanation [Southwick, 1991]. In the second type, however, the system has to bear the burden of interpreting the user's questions in the context of ongoing interaction and have multiple strategies to provide responses that satisfy different users [Swartout and Moore, 1993]. In the system we have designed the type of explanation provided is cooperative or proactive. Therefore, another area of evaluation would be testing whether the users might prefer reactive explanation in practical reasoning domain. Hypothesis such as "H2: Users will perceive reactive explanation to be more useful than proactive explanation." or the other way around "H3: Users will perceive proactive explanation to be more useful than reactive explanation." can be tested. How to generate reactive explanation in the first place is a topic we discuss as a part of future work (page 150).

6.4 Summary

In Chapter 2 (page 47), we argued in favour of argumentation-based approaches to practical reasoning and decision-making by stating arguments from literature about inadequacy of decision-theoretic models in certain cases and also by pointing out the explainability and justifiability of argumentation-based approaches. The latter being crucial in facilitating human understanding of a computational system has underpinned utilising argumentation in many areas such as practical reasoning, decision-making and planning [Amgoud and Prade, 2009; Gasque, 2013; Oren, 2013; Rahwan and Amgoud, 2006; Toniolo et al., 2011; Zhong et al., 2014]. Pursuing the same intention, in the previous chapter we proposed an approach to conducting normative practical reasoning using argumentation. However, in contrast to existing approaches to practical reasoning, in this chapter we exploited both aspects of argumentation for agent reasoning and agent dialogue, thereby providing a clear explanation of normative practical reasoning in natural language. In doing so, we used Caminada's dialogue for the preferred semantics in terms of a Socratic discussion. We showed that the plan arguments that are a member of a preferred extension can survive this type of discussion and therefore be justified dialectically, which make this work the first application of the preferred dialogues.

Chapter 7

Conclusions and Future Work

In this research, we have developed an effective approach that enables an autonomous agent to reason about its practical attitudes (i.e. actions, goals, norms) in search of the best course of actions to execute. Reasoning about what to do when the agent has multiple goals is a challenging task, particularly when these goals are conflicting. Moreover, social agents are often subject to norms imposed on them from the environment they operate in. The role of these norm is to regulate the autonomous agents behaviour in accordance with what is expected of them from the society's perspective. However, these norms may not be aligned with the agent's goals. The norms themselves can also be inconsistent. In such a complex environment the agent is not likely to be able to satisfy all its goals, while complying with all the norms imposed on it. What is required of a rational agent is to be able to reason about what to do with respect to its individual goals, the social norms and their conflicts, before committing to any course of actions. The sophistication of such reasoning mechanism, however, makes it very difficult to explain the reasoning process and its outcomes to a human user. In general, while intelligent agents can outperform humans computationally, the systems using them are not seen as transparent and hence trustworthy.

Argumentation has proven to be a promising paradigm for modelling common-sense and human-like reasoning [Bench-Capon and Dunne, 2007; Besnard and Hunter, 2008; Chesñevar et al., 2012]. It has also been expressed as a process of generating explanation in different domains [Caminada et al., 2014c; Fan and Toni, 2015; García et al., 2013; Schulz and Toni, 2014]. Therefore, as a paradigm, argumentation makes an attractive choice for handling the opacity arising from sophisticated operations in intelligent systems, including practical reasoning systems. Previous work on argumentation-based practical reasoning has focused on argumentation as a reasoning tool to determine the consistency of some plans [Ferrando et al., 2011; Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006; Tang and Parsons, 2005;

Toniolo, 2013] or discuss and compare a set of pre-generated plans to identify the best plan [Belesiotis et al., 2009; Gasque, 2013; Oren, 2013]. The dialogical aspect of argumentation in a multi-agent setting has been used either in order to conduct a dialogue that facilitates constructing a consistent plan, or to conduct a dialogue that facilitates reaching an agreement about what plan to execute. However, the role that argumentation-based dialogues can play in generating explanation for practical reasoning in complex domains is not explored to date.

The main objective of the research presented in this thesis was to set out an end-to-end solution to practical reasoning that takes an agent's actions as input and equips the agent with the ability to act in the presence of conflicting goals and norms, while also allowing the agent to explain why it acted as it did. The explanatory ability of argumentation has been used in some domains [Caminada et al., 2014c; Fan and Toni, 2015; Schulz and Toni, 2014] in the past, however it has not been used in practical reasoning. In order to address this objective, we have proposed a solution that integrates and extends several existing techniques in argumentation theory to offer an end-to-end process in which the following two questions are answered in a unified manner: (i) how should an agent act in a normative environment while it has conflicting goals and norms, and (ii) how can the agent explain to a human user why it acted in a certain way? In consequence, we:

- propose a formal model that formulates different types of conflicts and defines plans for the agent with respect to these conflicts;
- implement the model to obtain conflict-free plans;
- use argumentation frameworks to reason about the plans and their justifiability;
- propose a decision criterion to identify the best plan out of the set of justified plans that takes the preferences between goals and norms into account;
- use argumentation-based dialogues to explain the justifiability of plans in natural language.

The contributions of this work are explained further in the next section. Section 7.2 discusses the limitations of work presented alongside proposed solutions to tackle them. Some interesting further developments are discussed as future work in Section 7.3.

7.1 Contributions

This research proposes a framework for integrating practical and normative reasoning. The focus of many existing approaches to normative practical reasoning [Broersen et al., 2001;

Criado et al., 2010; Kollingbaum and Norman, 2003] has been answering the question of how should an agent act in a normative environment while it has conflicting goals and norms. We extend these approaches by answering this question using argumentation, such that the reasoning toward what to do is transparent and also explicable in a natural way.

While argumentation-based approaches to practical reasoning have been proposed in the past [Atkinson and Bench-Capon, 2007b; Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006], there has not been much attention paid to the normative and explanation aspect of an agent conducting practical reasoning in these frameworks. We extend these approaches by:

Putting norms at the heart of practical reasoning. In doing so, we have, for the first time, defined norm compliance and violation in the presence of durative actions. In addition, we have defined the conflict between norms in a novel way that takes the temporal essence of conflict between norms into account. Existing approaches, with the exception of a very recent work [Criado et al., 2015], treat conflict between norms statically by using a predefined preference order that determines which norm should be followed in case two norms are inconsistent. However, in dynamic environments, it can be quite challenging to specify all inconsistencies that may occur. In addition, as it is in this work, considering the temporal aspect of conflict, the inconsistency between norms can differ from one plan to another one. Two norms that are in conflict in the context of plan π do not necessarily have to be in conflict in the context of π'. In other words, it is the sequence of actions in the plan that clarifies whether two activated norms are in conflict in the plan. This type of inconsistency is impossible to predict prior to plan generation, and hence the inadequacy of existing approaches that pre-detect the conflicts between norms.

Considering planning as an inevitable part of practical reasoning. Plans defined by our formal model give a full temporal account of what actions should be executed, and in what order (e.g. concurrently, sequentially). Although planning is an inevitable part of practical reasoning, it has been abstracted away in several examples of existing literature [Amgoud et al., 2008a; Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006]. The main focus of these works is on identifying a subset of consistent desires and their plans. However, it is not clear how, i.e. when and in which orders, the agent should execute those plans. More importantly and as it is discussed by Bench-Capon and Atkinson [2009], it is difficult to distinguish between states and actions in these approaches, which in the end does not leave the agent with a clear choice of actions. However, practical reasoning is about what to do, rather than what to pursue. In contrast, the approaches [Atkinson and Bench-Capon, 2007b; Oren, 2013] that use AATS [Wooldridge and Hoek, 2006] for representing the agents actions, clearly distinguish between actions and states, but to the best of our knowledge, AATSs are merely used for knowledge representation purposes and have not been implemented. We extend the former category [Amgoud et al., 2008a; Hulstijn and van der Torre, 2004; Rahwan and Amgoud, 2006] of practical reasoning frameworks by giving a clear account of actions and states obtained from the execution of actions. The latter category [Atkinson and Bench-Capon, 2007b; Oren, 2013] are extended by providing an implementation of the model in ASP that automates the plan generation. The formulation of the model and the computational translation are very similar, thus there are no conceptual gaps to bridge.

Developing argumentation schemes for justifiability. The argument schemes in this work are developed in the spirit of Atkinson and Bench-Capon [2007b]. However, those schemes focus on actions rather than plans. So also the scheme in Toniolo [2013]. The schemes proposed in our work focus on plans and are mainly aimed at checking the justifiability of a plan with respect to agent's goals, norms, their preferences and conflicts. Hence, the accessibility of arguments in the dialogue for verifying the justifiability of a plan. Oren [2013] and Gasque [2013] also define argumentation schemes for plans, however the justifiability of plans with respect to a certain semantics is not addressed in any of them. The focus of the schemes in Oren [2013] is on defining preferences over paths or plans, whereas Gasque [2013] concentrates on the possibility of the plan with respect to an initial state and the values plans promote.

Using a novel decision criterion for identifying the best plan. The decision criterion identifies the best sequence of actions both in presence and absence of preference information over goals and norms. Preferences explicitly influence the justifiability of a plan by identifying certain attacks as defeat. Since all the justified plans are better than those not justified, the decision criterion is guaranteed to respect the preference information available. In addition, the preferences can be used to determine the norm- or goal-dominance of one plan over another one. However, the preferences over goals and norms are partial and there might be situations such that plans cannot be compared merely based on preferences. In such cases we use the number of goals satisfied and norms violated to compare justified plans. Depending on agent choice, a higher number of goals can take priority over a lower number of violations or the other way around.

Generating explanation for the best plan in natural language. The idea of using a dialogue to explain the agent's decision on the courses of actions is mentioned as a part of future work by Rahwan and Amgoud [2006] and Oren [2013]. This idea is fulfilled in this thesis. The explanation for comparison of justified plans is generated using a basic algorithm, in which the explanation of justifiability of the best plan is generated by translating a dialogue game for the preferred semantics. This is the first application of dialogue games for the preferred semantics in general, and Socratic discussion in particular. As discussed in Chapter 6 (page 141), due to the impracticality of the concept of restricted rebut, and despite the popularity of the preferred semantics, the dialogues developed for them are not yet used in practice.

7.2 Limitations

We plan to address some limitations of our current work in the future.

Enriching the conflict detection In Chapter 3 we explained three categories of conflicts, namely (i) conflict between goals, (ii) conflict between goals and norms, and (iii) conflict between norms. We also observed that the first two categories of conflict are static, while the last one is dynamic (i.e. contextual) and depends on the sequence of actions in which the norms are activated. Just like norms, in real scenarios, goals often have temporal properties, in terms of a deadline before which they should be satisfied, or a period over which they should stay satisfied (i.e. maintenance goals [Hindriks and Riemsdijk, 2007]). Temporally extended goals [Hindriks et al., 2009] are discussed in detail in agent programming languages such as GOAL [Boer et al., 2007], however they are not commonly used in practical reasoning frameworks, despite the importance of goals in these frameworks. Using temporally extended goals in the formal model instead of achievement goals, not only increases the expressiveness of the model, it also has considerable implications for conflict within goals and between goals and norms. It allows defining conflict within goals and between goals and norms contextually and thus enriching the concept of conflict in the model further. For instance, two temporal goals are defined as inconsistent if their requirements are inconsistent and the period over which they should be satisfied or stay satisfied overlaps. The conflict between goals and norms can also be defined in the same manner. An obligation and a goal are in conflict if the postconditions of the obliged action are inconsistent with the goal requirements and they have to be brought about over the same period of time. Similarly, a prohibition and a goal are in conflict if the postconditions of the prohibited action contribute to the goal requirements, however the norm forbids bringing them about over the same period over which the goal has to be satisfied or stay satisfied.

Enriching normative reasoning In the current model norms are all action-based and specified in terms of obligations and prohibitions. The normative reasoning capability of the agent can be enhanced by allowing state-based and permission norms. State-based norms would permit the expression of obligations and prohibitions in terms of achieving or avoiding some state before some deadline. A combination of action and state based norms (e.g. [De Vos et al., 2013]) enriches the norm representation as well as normative reasoning.

In addition permission norms are useful to model permitted violation of an obligation or a prohibition [Oren, 2013; Oren et al., 2010; Pacheco, 2012]. A temporal permission would permit the violation of an obligation or a prohibition over a certain period of time. The violations occurring in a period rather than the one covered by a permission norm, therefore remain undesirable. For example, off-street parking is often forbidden in central part of some cities apart from 7pm to 6am. In other words, 7pm to 6am is a period over which the prohibition of parking can be violated in the light of a permission norm.

Implementation Another area for improvement is to integrate the implementation of the normative practical reasoning model in ASP with the ASP implementation of ASPARTIX[1]. ASPARTIX is one of the most well-known implementations for computing various extensions of an abstract argumentation framework. The goal of such an integration would be improving the efficiency of the system in comparison with the current version, in which the argumentation frameworks for plans has to be queried about whether they entail a plan argument in at least one preferred extension.

To make such an integration, we propose using one of the newest developments in the world of ASP solvers: *multi-shot* solving [Gebser et al., 2015] with *clingo* 4. Multi-shot solving allows running and solving multiple models concurrently. In fact, one can solve more than one answer set program concurrently by creating and calling different instances of the solver. It is also possible to run answer set programs sequentially, when the answers of the first program need to be extracted and re-inserted to the second one for further processing.

Assuming the implementation of the planning problem in Chapter 4 is the first answer set program, and ASPARTIX code for computing the preferred extensions of an argumentation framework is the second answer set program, it seems plausible to extract arguments for plans, goals and norms from the first and insert it into the second, hence computing if

[1]Accessible at: http://www.dbai.tuwien.ac.at/proj/argumentation/systempage/ .

a plan is credulously accepted under the preferred semantics through a unified process.

7.3 Future Work

With respect to future work, we see several promising avenues to extend the work presented in this thesis:

Institutionally situated practical reasoning agent Norms in this work are regulations imposed on the agent externally, however the normative frameworks that enforce these norms are not modelled. Institutions provide an effective mechanism to govern agent's behaviour by specifying a set of norms that fulfil certain normative objectives [Cliffe, 2007; Vázquez-Salceda, 2003]. Norms imposed on the agent might be from several institutions and hence there might be conflict between them. In a bigger picture, a practical reasoning agent's actions are affected by institutions that the agent subscribes and each institution imposes a certain set of norms on the agent that the agent might comply with or violate. The benefit of reasoning about norms in the context of institutions is in detecting the normative conflict in a more systematic manner and at the institution level rather than instantiated norm level. There is a large body of work available on conflict-free institutions that can be used by agents to try to subscribe to institutions that are likely to cause less conflict where possible. Fortunately, reasoning about institutions, including checking their interactions and merging outcome is formulated in ASP [Lee et al., 2013; Li, 2014; Li et al., 2013] which should make it easier to integrate with our current implementation of a practical reasoning agent in ASP.

Another potential of making such an extension is to consider the legitimacy of a plan from a specific institution's perspective. In this case the institutions can be seen as audiences and plans are evaluated from the perspective of various audiences. A plan might not violate any norm of a particular institution, while it violates several norms of another. This ultimately makes the plan legitimate from perspective of the former institution, while the latter might question the legitimacy of the plan. As a result, a practical reasoning agent can view and compare its plans from its audiences' point of view.

Human-agent interaction The normative practical reasoning agent in this work is able to explain its decision about the best plan to a human user using algorithms that translate to natural language the justifiability of the best plan and its superiority to other plans. The explanation is presented to the user so that the user can scrutinise and understand the system and its outcome. However, the user does not interact with the system. Ideally the user should be able to query the agent about why a plan is justified or not. Also the queries about why a plan does not satisfy a goal or violates a norm can be put forward by the user.

The justifiability of a plan is explained using a dialogue for the preferred semantics. Currently, there is no implementation of human-agent dialogue for the preferred semantics. An implementation of a user-computer dialogue for the grounded semantics is proposed in [Caminada and Podlaszewski, 2012a] [2]. The dialogue is based on argument labellings and although is not linked to natural language generation, it is intuitive enough to be used in a user-computer setting. Caminada et al. [2014c] have recently developed another demonstrator that uses dialogical proof procedure based on the grounded semantics, with the purpose of allowing a user to identify the reasons behind a claim being accepted under the grounded semantics. Unlike Caminada and Podlaszewski [2012a], Caminada et al. [2014c] uses a natural language generation unit that makes the dialogue accessible to non-technical users. An interesting line of future work would be to implement a similar demonstrator for the preferred semantics that is used by a human user to query the agent about why a plan is justified.

Applications in collaborative planning Norms are a well understood approach for declaratively specifying desirable behaviour in multi-agent systems, and are often used to drive cooperation and coordination [Governatori et al., 2011; Jonker et al., 2007; Oren et al., 2011; Pacheco, 2012]. They have therefore been used extensively in collaborative planning in multi-agent systems [Balke et al., 2013; Oh et al., 2011; Vasconcelos et al., 2009]. Argumentation-based approaches to collaborative planning [Black and Atkinson, 2009; Black and Bentley, 2011; Ferrando et al., 2011; Gasque, 2013; Toniolo et al., 2012] have successfully established to enhance the collaboration manner and reaching agreement between a team of agents by exchanging arguments. However, these approaches, with the exception of Toniolo [2013] do not consider norm-driven plans. The consideration of norms in Toniolo [2013] itself is limited to a simple notion in situation calculus. Furthermore, there is no evidence of reasoning about norms, as it is understood in the normative literature, in terms of norm compliance or violation; the agents have to comply with norms imposed on them, without possibility of violating them, which practically is norm regimentation. The approach we presented in this thesis clearly gives the choice of norm violation or compliance to the agent, however it is developed for a single-agent setting. Given the proven power of norms in collaborative planning and the benefit of argumentation in this area, another line of future work would be to extend the approach presented

[2]See https://code.google.com/p/pyafl/downloads/list for implementation.

for normative practical reasoning presented in this work, to multi-agent norm-driven collaborative planning.

Applications in decision support systems In recent years, there has been an increasing number of applications of intelligent agents in decision support systems [Atkinson et al., 2004, 2006; Bench-Capon et al., 2012; Glasspool et al., 2006]. Agents are capable of outperforming humans in many complicated activities including decision-making. Cognitive limitations of human decision-makers (e.g. ignoring probabilities, preferences and irrelevant generalisations) are often recognised as the main source of *irrational* decision-making [Axelrod, 1976; Nisbett and Ross, 1980]. While decision support systems are free from these difficulties, they have proven to be difficult for decision-makers to trust [Introne and Iandoli, 2014; Lawrence et al., 2002]. There is experimental evidence that argumentation-based decision support systems increase the trustworthiness of intelligent agents decision-makers [Introne and Iandoli, 2014; Paul A., 2003; Scheuer et al., 2010]. Familiarity of most people with argumentation as a skill and also the availability of graphical representation of arguments and argumentation-based decision-making are identified as the main reasons for this evidence [Mercier and Sperber, 2011; Toth et al., 2002].

The argumentation-based normative practical reasoning agent modelled in this thesis can be used to support the decision-making process for various users with different preferences over goals and norms and different priorities over goal- and norm-dominance. Currently the agent identifies the best plan based on a pre-defined set of preferences over goals and norms. The decision criterion that identifies the best decision is also pre-defined. However, in a human-to-agent settings, these elements should be defined dynamically by the users. Similarly, the users should be capable of querying the agent about the best decision based on different decision criterions. One user might define the best decision as a decision with most pros, while another user might define it as a decision with least cons. The agent on the other hand, should be able to present the pros and cons of each decision in terms of arguments, making it easy for the user to understand the advantages and disadvantages of each decision with respect to different decision citerions.

Empirical evaluation of the explanation generation In the previous chapter (page 142) we discussed the difference between proactive and reactive explanation generation. An interesting area of future work would be to empirically verify which type of explanation generation is more useful in practical reasoning domain. This is of course subject to the implementation of reactive explanation generation in the first place, which itself is a part of future work discussed earlier in this section. Moreover, the empirical evaluation can consider novice as well as expert users and test if different types of explanations might be more useful to different types of users with various level of expertise. A study of such nature was conducted by Ye [1995], however, in this study novice and expert users are tested to verify whether they differ in valuing alternative explanations being provided in terms of (i) the trace of inference rules used to reach a certain conclusion; (ii) the justification of a reached conclusion; (iii) the overall strategy of the system to reach a conclusion. But this study can nevertheless be useful to outline a study in which different types of explanation in terms of one-way (proactive explanation generation) two-ways (reactive explanation generation) are compared based on their usefulness for novice and expert users.

Appendix A

ASP Code of Example

```
%States
state(0..20).

%Initial state
holdsat(employee,0).
holdsat(given_birth,0).

%Inertial fluents
holdsat(X,S2) :- holdsat(X,S1), not terminated(X,S1), state(S1;S2),
                 S2=S1+1.

%Actions with their pre and postcoditions
action(join_union,1).
pre(join_union,S) :-  not holdsat(union_member,S),state(S).
holdsat(union_member,S2) :- executed(join_union,S1),
        action(join_union,1), state(S1;S2), S2=S1+1.

action(finish_report,3).
pre(finish_report,S) :-  holdsat(at_office,S),state(S).
holdsat(report_finalised,S2) :- executed(finish_report,S1),
            action(finish_report,3), state(S1;S2), S2=S1+3.

action(go_to_office,1).
pre(go_to_office,S) :- not holdsat(office,S),state(S).
holdsat(at_office,S2) :- executed(go_to_office,S1),
        action(go_to_office,1), state(S1;S2), S2=S1+1.
action(attend_interview,4).
pre(attend_interview,S) :- holdsat(theory_test_done,S),
      holdsat(at_interview_venue,S),not holdsat(at_office,S),
                    not holdsat(at_meeting_venue,S),state(S).
holdsat(interviewed,S2) :- executed(attend_interview,S1),
       action(attend_interview,4), state(S1;S2), S2=S1+4.
terminated(at_interview_venue,S2) :- executed(attend_interview,S1),
                 action(attend_interview,4), state(S1;S2), S2=S1+3.

action(take_theory_test,1).
pre(take_theory_test,S) :- holdsat(course_fee_paid,S),state(S).
holdsat(theory_test_done,S2) :- executed(take_theory_test,S1),
            action(take_theory_test,1), state(S1;S2), S2=S1+1.

action(get_company_funding,2).
pre(get_company_funding,S) :-  holdsat(employee,S), state(S).
holdsat(course_fee_paid,S2) :- executed(get_company_funding,S1),
           action(get_company_funding,2), state(S1;S2), S2=S1+2.

action(attend_meeting,5).
pre(attend_meeting,S) :-  holdsat(course_fee_paid,S),
    holdsat(at_meeting_venue,S), not holdsat(at_interview_venue,S),
                                 not holdsat(at_office,S),state(S).
holdsat(meeting_attended,S2) :- executed(attend_meeting,S1),
                   action(attend_meeting,5), state(S1;S2), S2=S1+5.
holdsat(summary_documented,S2) :- executed(attend_meeting,S1),
                   action(attend_meeting,5), state(S1;S2), S2=S1+5.
terminated(at_meeting_venue,S2) :- executed(attend_meeting,S1),
                   action(attend_meeting,5), state(S1;S2), S2=S1+4.

action(go_to_interview_venue,1).
pre(go_to_interview_venue,S) :- not holdsat(at_interview_venue,S),
                                                         state(S).
holdsat(at_interview_venue,S2) :-
                               executed(go_to_interview_venue,S1),
           action(go_to_interview_venue,1), state(S1;S2), S2=S1+1.
terminated(at_office,S) :- executed(go_to_interview_venue,S),
                   action(go_to_interview_venue,1), state(S).
terminated(at_meeting_venue,S) :- executed(go_to_interview_venue,S),
                          action(go_to_interview_venue,1), state(S).
action(go_to_meeting_venue,1).
pre(go_to_meeting_venue,S) :- not holdsat(at_meeting_venue,S),
                                                     state(S).
holdsat(at_meeting_venue,S2) :-
                              executed(go_to_meeting_venue,S1),
         action(go_to_meeting_venue,1), state(S1;S2), S2=S1+1.
terminated(at_office,S) :- executed(go_to_meeting_venue,S),
                   action(go_to_meeting_venue,1), state(S).
terminated(at_interview_venue,S) :- executed(go_to_meeting_venue,S),
                             action(go_to_meeting_venue,1), state(S).

action(take_maternity_leave,2).
pre(take_maternity_leave,S) :-  holdsat(given_birth,S),state(S).
holdsat(on_maternity_leave,S2) :- executed(take_maternity_leave,S1),
              action(take_maternity_leave,2), state(S1;S2), S2=S1+2.

%Action execution
{executed(A,S)} :- action(A,D), state(S).

%Inprogress actions
inprog(A,S2) :- executed(A,S1), action(A,D), state(S1;S2),
                                       S1 <= S2, S2< S1+D.

%Preserving preconditions during action execution
:- inprog(A,S), action(A,D), not pre(A,S), state(S).

%Preventing the execution of two actions in the same state
:- executed(A1,S), executed(A2,S), A1!=A2, action(A1,D1),
                                 action(A2,D2), state(S).

%Norm n1
holdsat(o(n1,attend_meeting,2+S2),S2) :-
                                executed(get_company_funding,S1),
             action(get_company_funding,2), S2=S1+2,state(S1;S2).
cmp(o(n1,attend_meeting,DL),S) :-
                              holdsat(o(n1,attend_meeting,DL),S),
                                      executed(attend_meeting,S),
                        action(attend_meeting,5),state(S), S!=DL.
complied(n1) :- cmp(o(n1,attend_meeting,DL),S),state(S).
terminated(o(n1,attend_meeting,DL),S) :-
                        cmp(o(n1,attend_meeting,DL),S), state(S).
```

```
vol(o(n1,attend_meeting,DL),S) :-
              holdsat(o(n1,attend_meeting,DL),S), DL=S, state(S).
violated(n1) :- vol(o(n1,attend_meeting,DL),S),state(S).
terminated(o(n1,attend_meeting,DL),S) :-
                        vol(o(n1,attend_meeting,DL),S), state(S).

%Norm n2
holdsat(f(n2,go_to_office,6+S2),S2) :-
                               executed(take_maternity_leave,S1),
           action(take_maternity_leave,2), S2=S1+2, state(S1;S2).
cmp(f(n2,go_to_office,DL),S) :- holdsat(f(n2,go_to_office,DL),S),
                          action(go_to_office,1), DL=S, state(S).
complied(n2) :- cmp(f(n2,go_to_office,DL),S),state(S).
terminated(f(n2,go_to_office,DL),S) :-
                          cmp(f(n2,go_to_office,DL),S), state(S).
vol(f(n2,go_to_office,DL),S) :- holdsat(f(n2,go_to_office,DL),S),
                       executed(go_to_office,S), state(S), S!=DL.
violated(n2) :- vol(f(n2,go_to_office,DL),S),state(S).
terminated(f(n2,go_to_office,DL),S) :-
                          vol(f(n2,go_to_office,DL),S), state(S).

%Norm n3
holdsat(f(n3,attend_meeting,6+S2),S2) :-
                               executed(take_maternity_leave,S1),
           action(take_maternity_leave,2), S2=S1+2, state(S1;S2).
cmp(f(n3,attend_meeting,DL),S) :-
                              holdsat(f(n3,attend_meeting,DL),S),
                        action(attend_meeting,5), DL=S, state(S).
complied(n3) :- cmp(f(n3,attend_meeting,DL),S),state(S).
terminated(f(n3,attend_meeting,DL),S) :-
                        cmp(f(n3,attend_meeting,DL),S), state(S).
vol(f(n3,attend_meeting,DL),S) :-
                              holdsat(f(n3,attend_meeting,DL),S),
                     executed(attend_meeting,S), state(S), S!=DL.
violated(n3) :- vol(f(n3,attend_meeting,DL),S),state(S).
terminated(f(n3,attend_meeting,DL),S) :-
                        vol(f(n3,attend_meeting,DL),S), state(S).
```

```
%Norm n4
holdsat(o(n4,attend_interview,2+S2),S2) :-
                                   executed(take_theory_test,S1),
                action(take_theory_test,1), S2=S1+1,state(S1;S2).
cmp(o(n4,attend_interview,DL),S) :-
                            holdsat(o(n4,attend_interview,DL),S),
                                   executed(attend_interview,S),
                      action(attend_interview,4),state(S), S!=DL.
complied(n4) :- cmp(o(n4,attend_interview,DL),S),state(S).
terminated(o(n4,attend_interview,DL),S) :-
                      cmp(o(n4,attend_interview,DL),S), state(S).
vol(o(n4,attend_interview,DL),S) :-
            holdsat(o(n4,attend_interview,DL),S), DL=S, state(S).
violated(n4) :- vol(o(n4,attend_interview,DL),S),state(S).
terminated(o(n4,attend_interview,DL),S) :-
                      vol(o(n4,attend_interview,DL),S), state(S).

%Goals
satisfied(strike,S) :- not holdsat(at_office,S),
                                 not holdsat(meeting_attended,S),
                               holdsat(union_member,S), state(S).
satisfied(strike) :- satisfied(strike,S), state(S).

satisfied(submission,S) :- holdsat(at_office,S),
                           holdsat(report_finalised,S), state(S).
satisfied(submission) :- satisfied(submission,S), state(S).

satisfied(certificate,S) :- holdsat(course_fee_paid,S),
    sholdsat(theory_test_done,S), holdsat(interviewed,S),state(S).
satisfied(certificate) :- satisfied(certificate,S), state(S).

%Preventing sequence that do not satisfy any of the goals
:- not satisfied(strike), not satisfied(submission),
                         not satisfied(certificate).

%Conflicting goals
:- satisfied(strike), satisfied(submission).

%Conflicting actions
:- inprog(finish_report,S), inprog(go_to_office,S),
         action(finish_report,3), action(go_to_office,1), state(S).
```

```
:- inprog(finish_report,S), inprog(attend_interview,S),
     action(finish_report,3), action(attend_interview,4), state(S).
:- inprog(finish_report,S), inprog(attend_meeting,S),
       action(finish_report,3), action(attend_meeting,5), state(S).
:- inprog(finish_report,S), inprog(go_to_interview_venue,S),
 action(finish_report,3), action(go_to_interview_venue,1), state(S).
:- inprog(finish_report,S), inprog(go_to_meeting_venue,S),
   action(finish_report,3), action(go_to_meeting_venue,1), state(S).
:- inprog(go_to_office,S),  inprog(attend_interview,S),
      action(go_to_office,1), action(attend_interview,4), state(S).
:- inprog(go_to_office,S), inprog(attend_meeting,S),
        action(go_to_office,1), action(attend_meeting,5), state(S).
:- inprog(go_to_office,S), inprog(go_to_interview_venue,S),
 action(go_to_office,1), action(go_to_interview_venue,1), state(S).
:- inprog(go_to_office,S), inprog(go_to_meeting_venue,S),
   action(go_to_office,1), action(go_to_meeting_venue,1), state(S).
:- inprog(attend_interview,S), inprog(attend_meeting,S),
    action(attend_interview,4), action(attend_meeting,5), state(S).
:- inprog(attend_interview,S), inprog(go_to_interview_venue,S),
                                    action(attend_interview,4),
                     action(go_to_interview_venue,1), state(S).
:- inprog(attend_interview,S), inprog(go_to_meeting_venue,S),
                                  action(attend_interview,4),
                     action(go_to_meeting_venue,1), state(S).
:- inprog(attend_meeting,S), inprog(go_to_interview_venue,S),
                                    action(attend_meeting,5),
                   action(go_to_interview_venue,1), state(S).
:- inprog(attend_meeting,S), inprog(go_to_meeting_venue,S),
                                  action(attend_meeting,5),
                   action(go_to_meeting_venue,1), state(S).
:- inprog(go_to_interview_venue,S), inprog(go_to_meeting_venue,S),
                                  action(go_to_interview_venue,1),
                          action(go_to_meeting_venue,1), state(S).

%Conflicting goals and norms
:- satisfied(strike), complied(n1).
:- satisfied(submission), complied(n2).
```

```
%Optimisation rules

%Preventing the execution of an action more than once
:- executed(A,S1),executed(A,S2), S1!=S2, state(S1;S2), action(A,D).

%Detecting the final state
flag(S2) :- satisfied(strike,S2), not satisfied(strike,S1),
                                      state(S1;S2), S2=S1+1.
flag(S2) :- satisfied(submission,S2), not satisfied(submission,S1),
                                              state(S1;S2), S2=S1+1.
flag(S2) :- satisfied(certificate,S2),not satisfied(certificate,S1),
                                               state(S1;S2), S2=S1+1.
final(F) :- F= #max[flag(S)=S].

%No idle state before the final state
alpha(S) :- inprog(A,S), action(A,D), state(S).
:- final(S2), not alpha(S1), state(S1;S2), S1<S2.

%No action in progress after the final state
:- final(S1), alpha(S2), state(S1;S2),  S2 >= S1.
```

Appendix B

Proof of Properties in Chapter 5

Proof of Property 5.1. We need to show: $\forall \pi \in \Pi$, $(\pi, \pi) \notin >_\pi$.
π cannot be justified and unjustified at the same time. Also, both $>_G$ and $>_N$ are irreflexive so $(\pi, \pi) \notin >_G$, and $(\pi, \pi) \notin >_N$. Thus, $(\pi, \pi) \notin >_\pi$.

Proof of Property 5.2. We need to show: $\forall \pi_1, \pi_2 \in \Pi$, if $\pi_1 \neq \pi_2$ and $(\pi_1, \pi_2) \in >_\pi$ then $(\pi_2, \pi_1) \notin >_\pi$.
Assume that $\pi_1 \neq \pi_2$ and $(\pi_1, \pi_2) \in >_\pi$ while $(\pi_2, \pi_1) \in >_\pi$.

case 1: $(\pi_2, \pi_1) \in >_\pi$ because π_2 is justified and π_1 is not. This means that $(\pi_1, \pi_2) \notin >_\pi$ which is contrary to assumption.

case 2: $(\pi_2, \pi_1) \in >_\pi$ because π_2 and π_1 are both justified and $(\pi_2, \pi_1) \in >_G$. Since $>_G$ is antisymmetric $(\pi_1, \pi_2) \notin >_G$, which means that $(\pi_1, \pi_2) \notin >_\pi$ that is contrary to assumption.

case 3: $(\pi_2, \pi_1) \in >_\pi$ because π_2 and π_1 are both justified and $(\pi_2, \pi_1) \in \sim_G$ but $(\pi_1, \pi_2) \in >_N$. Since $\sim_G$ is symmetric, $(\pi_1, \pi_2) \in \sim_G$ and since $>_N$ is antisymmetric $(\pi_2, \pi_1) \notin >_N$, which means that $(\pi_1, \pi_2) \notin >_\pi$ that is contrary to assumption.

Proof of Property 5.3. We need to show: $\forall \pi_1, \pi_2, \pi_3 \in \Pi$, if $(\pi_1, \pi_2) \in >_\pi$ and $(\pi_2, \pi_3) \in >_\pi$, then $(\pi_1, \pi_3) \in >_\pi$.

case 1: $(\pi_1, \pi_2) \in >_\pi$ because π_1 is justified but π_2 is not. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 is justified but π_3 is not.
This cannot be the case since π_2 cannot be justified and unjustified at the same time.

case 2: $(\pi_1, \pi_2) \in >_\pi$ because π_1 is justified but π_2 is not. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 and π_3 are both justified but $(\pi_2, \pi_3) \in >_G$.
This cannot be the case since π_2 cannot be justified and unjustified at the same time.

case 3: $(\pi_1, \pi_2) \in >_\pi$ because π_1 is justified but π_2 is not. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 and π_3 are both justified and $(\pi_2, \pi_3) \in \sim_G$, while $(\pi_3, \pi_2) \in >_N$.
This cannot be the case since π_2 cannot be justified and unjustified at the same time.

case 4: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified but $(\pi_1, \pi_2) \in >_G$. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 is justified but π_3 is not.
Since π_1 is justified and π_3 is not, $(\pi_1, \pi_3) \in >_\pi$.

case 5: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified but $(\pi_1, \pi_2) \in >_G$. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 and π_3 are both justified but $(\pi_2, \pi_3) \in >_G$.
Since $>_G$ is transitive, from $(\pi_1, \pi_2) \in >_G$ and $(\pi_2, \pi_3) \in >_G$ we conclude that $(\pi_1, \pi_3) \in >_G$. Thus, $(\pi_1, \pi_3) \in >_\pi$.

case 6: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified but $(\pi_1, \pi_2) \in >_G$. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 and π_3 are both justified and $(\pi_2, \pi_3) \in \sim_G$ and but $(\pi_3, \pi_2) \in >_N$.
Since $>_G$ and $\sim_G$ are both transitive, from $(\pi_1, \pi_2) \in >_G$ and $(\pi_2, \pi_3) \in \sim_G$ we conclude that $(\pi_1, \pi_3) \in >_G$. Thus, $(\pi_1, \pi_3) \in >_\pi$.

case 7: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified and $(\pi_2, \pi_1) \in \sim_G$ but $(\pi_1, \pi_2) \in >_N$. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 is justified but π_3 is not.
Since π_1 is justified and π_3 is not, $(\pi_1, \pi_3) \in >_\pi$.

case 8: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified and and $(\pi_2, \pi_1) \in \sim_G$ but $(\pi_1, \pi_2) \in >_N$. On the other hand, $(\pi_2, \pi_3) \in >_\pi$ because π_2 and π_3 are both justified but $(\pi_2, \pi_3) \in >_G$.
Since $>_G$ and $\sim_G$ are both transitive, from $(\pi_1, \pi_2) \in \sim_G$ and $(\pi_2, \pi_3) \in >_G$ we conclude that $(\pi_1, \pi_3) \in >_G$. Thus, $(\pi_1, \pi_3) \in >_\pi$.

case 9: $(\pi_1, \pi_2) \in >_\pi$ because π_1 and π_2 are both justified and $(\pi_1, \pi_2) \in \sim_G$ but $(\pi_2, \pi_1) \in >_N$. On the other hand, $(\pi_3, \pi_2) \in >_\pi$ because π_2 and π_3 are both justified and and $(\pi_2, \pi_3) \in \sim_G$ and but $(\pi_3, \pi_2) \in >_N$.
Since $\sim_G$ is transitive, from $(\pi_1, \pi_2) \in \sim_G$ and $(\pi_2, \pi_3) \in \sim_G$ we conclude that $(\pi_1, \pi_3) \in \sim_G$. Also since Since $>_N$ is transitive, from $(\pi_3, \pi_2) \in >_N$ and $(\pi_2, \pi_1) \in >_N$ we conclude that $(\pi_3, \pi_1) \in >_N$. Thus, $(\pi_1, \pi_3) \in >_\pi$.

Proof of Property 5.4. We need to show that $\sim_\pi$ is reflexive, symmetric and transitive.

Reflexive: Assume that $\exists \pi \in \Pi$ s.t. $(\pi, \pi) \notin \sim_\pi$. If $(\pi, \pi) \notin \sim_\pi$ then $(\pi, \pi) \in >_\pi$, which cannot be the case since $>_\pi$ is not reflexive.

Symmetric: Assume that $(\pi_1, \pi_2) \in \sim_\pi$ but $(\pi_2, \pi_1) \notin \sim_\pi$. If $(\pi_2, \pi_1) \notin \sim_\pi$ then either

case 1: $(\pi_1, \pi_2) \in >_\pi$, which means that $(\pi_1, \pi_2) \notin \sim_\pi$. This is contrary to assumption $(\pi_1, \pi_2) \in \sim_\pi$. Therefore, $(\pi_2, \pi_1) \in \sim_\pi$.

case 2: $(\pi_2, \pi_1) \in >_\pi$, which means that $(\pi_1, \pi_2) \notin \sim_\pi$. This is contrary to assumption $(\pi_1, \pi_2) \in \sim_\pi$. Therefore, $(\pi_2, \pi_1) \in \sim_\pi$.

Transitive: We need to show that if $(\pi_1, \pi_2) \in \sim_\pi$ and $(\pi_2, \pi_3) \in \sim_\pi$, then $(\pi_1, \pi_3) \in \sim_\pi$.

case 1: $(\pi_1, \pi_2) \in \sim_\pi$ because they are both not justified. Also $(\pi_2, \pi_3) \in \sim_\pi$ because they are both not justified. Therefore, neither of π_1 and π_3 are justified, so $(\pi_1, \pi_3) \in \sim_\pi$.

case 2: $(\pi_1, \pi_2) \in \sim_\pi$ because they are both not justified. On the other hand, $(\pi_2, \pi_3) \in \sim_\pi$ because they are both justified and $(\pi_2, \pi_3) \in \sim_G$ and $(\pi_2, \pi_3) \in \sim_N$. But π_2 cannot be both justified and not justified at the same time.

case 3: $(\pi_1, \pi_2) \in \sim_\pi$ because they are both justified and $(\pi_1, \pi_2) \in \sim_G$ and $(\pi_1, \pi_2) \in \sim_N$. On the other hand, $(\pi_2, \pi_3) \in \sim_\pi$ because neither of them is justified. But π_2 cannot be both justified and not justified at the same time.

case 4: $(\pi_1, \pi_2) \in \sim_\pi$ because they are both justified and $(\pi_1, \pi_2) \in \sim_G$ and $(\pi_1, \pi_2) \in \sim_N$. On the other hand, $(\pi_2, \pi_3) \in \sim_\pi$ because they are both justified and $(\pi_2, \pi_3) \in \sim_G$ and $(\pi_2, \pi_3) \in \sim_N$. Since $\sim_G$ and $\sim_N$ are both transitive, it follows that π_1 and π_3 are both justified and $(\pi_1, \pi_3) \in \sim_G$ and $(\pi_1, \pi_3) \in \sim_N$. Thus, $(\pi_1, \pi_3) \in \sim_\pi$.

Proof of Property 5.5. $\geq$ is a total order on Π iff it is antisymmetric, transitive and total.

- antisymmetric: We need to prove that if $([\pi_i], [\pi_j]) \in \geq$ and $([\pi_j], [\pi_i]) \in \geq$ then $[\pi_i] = [\pi_j]$. If $([\pi_i], [\pi_j]) \in \geq$, then $(\pi_i, \pi_j) \in >_\pi$ or $(\pi_i, \pi_j) \in \sim_\pi$. If $(\pi_i, \pi_j) \in >_\pi$, then $(\pi_j, \pi_i) \notin >_\pi$. Because $([\pi_j], [\pi_i]) \in \geq$, we have to have $(\pi_j, \pi_i) \in \sim_\pi$, which leads to $[\pi_j] = [\pi_i]$.

- transitive: We need to show that if $([\pi_i],[\pi_j]) \in \geq$ and $([\pi_j],[\pi_i]) \in \geq$, then $([\pi_i],[\pi_k]) \in \geq$. If $([\pi_i],[\pi_j]) \in \geq$ then $(\pi_i,\pi_j) \in >_\pi$ or $(\pi_i,\pi_j) \in \sim_\pi$. Similarly, if $([\pi_j],[\pi_k]) \in \geq$ then $(\pi_j,\pi_k) \in >_\pi$ or $(\pi_j,\pi_k) \in \sim_\pi$. Since $>_\pi$ are $\sim_\pi$ are both transitive, in either of the following four cases we conclude that $([\pi_i],[\pi_k]) \geq$:

  case 1: $(\pi_i,\pi_j) \in >_\pi$, and $(\pi_j,\pi_k) \in >_\pi$ implies $(\pi_i,\pi_k) \in >_\pi$ and therefore $([\pi_i],[\pi_k)] \in \geq$.

  case 2: $(\pi_i,\pi_j) \in >_\pi$, and $(\pi_j,\pi_k) \in \sim_\pi$ implies $(\pi_i,\pi_k) \in >_\pi$ and therefore $([\pi_i],[\pi_k)] \in \geq$.

  case 3: $(\pi_i,\pi_j) \in \sim_\pi$, and $(\pi_j,\pi_k) \in >_\pi$ implies $(\pi_i,\pi_k) \in >_\pi$ and therefore $([\pi_i],[\pi_k)] \in \geq$.

  case 4: $(\pi_i,\pi_j) \in \sim_\pi$, and $(\pi_j,\pi_k) \in \sim_\pi$ implies $(\pi_i,\pi_k) \in \sim_\pi$ and therefore $([\pi_i],[\pi_k)] \in \geq$.

- total: If $\geq$ is not total, then $\exists[\pi_i],[\pi_j]$ s.t. $([\pi_i],[\pi_j]) \notin \geq$ and $([\pi_j],[\pi_i]) \notin \geq$. If $([\pi_i],[\pi_j]) \notin \geq$, then $(\pi_i,\pi_j) \notin >_\pi$ and $(\pi_i,\pi_j) \notin \sim_\pi$. On the other hand, if $([\pi_j],[\pi_i]) \notin \geq$, then $(\pi_j,\pi_i) \notin >_\pi$ and $(\pi_j,\pi_i) \notin \sim_\pi$. From $(\pi_i,\pi_j) \notin >_\pi$ and $(\pi_j,\pi_i) \notin >_\pi$ we conclude that $(\pi_i,\pi_j) \in \sim_\pi$, which is contradictory to $(\pi_i,\pi_j) \notin \sim_\pi$ and $(\pi_j,\pi_i) \notin \sim_\pi$.

Bibliography

Aker, Erdi, Volkan Patoglu, and Esra Erdem (2013). “Collaborative housekeeping robotics using Answer Set Programming”. In: *21st Signal Processing and Communications Applications Conference, SIU 2013, Haspolat, Turkey, April 24-26, 2013*. IEEE, pp. 1–4 (pages 76, 97).

Aldewereld, Huib, Frank Dignum, Andrés García-Camino, Pablo Noriega, Juan A. Rodríguez-Aguilar, and Carles Sierra (2006). “Operationalisation of norms for usage in electronic institutions”. In: *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*. Ed. by Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss, and Peter Stone. ACM, pp. 223–225 (page 28).

Alrawagfeh, Wagdi and Felipe Meneguzzi (2014). “Utilizing Permission Norms in BDI Practical Normative Reasoning”. In: *16th International Workshop on Coordination, Organizations, Institutions, and Norms* (pages 27, 77).

Amgoud, Leila (2003). “A Formal Framework for Handling Conflicting Desires”. In: *ECSQARU*. Ed. by Thomas D. Nielsen and Nevin Lianwen Zhang. Vol. 2711. Lecture Notes in Computer Science. Springer, pp. 552–563 (pages 15, 47, 48, 51, 52).

— (2012). “A unified setting for inference and decision: An argumentation-based approach”. In: *CoRR* abs/1207.1363 (pages 100, 114, 132).

Amgoud, Leila and Claudette Cayrol (2002). “A Reasoning Model Based on the Production of Acceptable Arguments”. In: *Annals of Mathematics Artificial Intelligence* 34.1-3, pp. 197–215 (pages 35, 102, 104, 105).

Amgoud, Leila and Henri Prade (2004a). “Reaching Agreement Through Argumentation: A Possibilistic Approach”. In: *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada, June 2-5,* *2004*. Ed. by Didier Dubois, Christopher A. Welty, and Mary-Anne Williams. AAAI Press, pp. 175–182 (page 36).

Amgoud, Leila and Henri Prade (2004b). “Using Arguments for Making Decisions: A Possibilistic Logic Approach”. In: *UAI ’04, Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence, Banff, Canada, July 7-11, 2004*. Ed. by David Maxwell Chickering and Joseph Y. Halpern. AUAI Press, pp. 10–17 (page 100).

— (2009). “Using arguments for making and explaining decisions”. In: *Artificial Intelligence* 173.3-4, pp. 413–436 (pages 100, 130, 143).

Amgoud, Leila and Srdjan Vesic (2009). “Repairing Preference-Based Argumentation Frameworks”. In: *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*. Ed. by Craig Boutilier, pp. 665–670 (page 104).

— (2010). “Handling Inconsistency with Preference-Based Argumentation”. In: *Scalable Uncertainty Management - 4th International Conference, SUM 2010, Toulouse, France, September 27-29, 2010. Proceedings*. Ed. by Amol Deshpande and Anthony Hunter. Vol. 6379. Lecture Notes in Computer Science. Springer, pp. 56–69 (page 100).

— (2012). “A formal analysis of the role of argumentation in negotiation dialogues”. In: *J. Log. Comput.* 22.5, pp. 957–978 (pages 32, 44).

— (2014). “Rich preference-based argumentation frameworks”. In: *Int. J. Approx. Reasoning* 55.2, pp. 585–606 (pages 103, 104, 118).

Amgoud, Leila, Claudette Cayrol, and Marie-Christine Lagasquie-Schiex (2004). “On the bipolarity in argumentation frameworks”. In: *10th International Workshop on Non-Monotonic Reasoning (NMR)*. Ed. by James P. Delgrande and Torsten Schaub, pp. 1–9 (pages 15, 35, 36).

Amgoud, Leila, Caroline Devred, and Marie-Christine Lagasquie-Schiex (2008a). “A Constrained Argumentation System for Practical Reasoning”. In: *ArgMAS*. Ed. by Iyad Rahwan and Pavlos Moraitis. Vol. 5384. Lecture Notes in Computer Science. Springer, pp. 37–56 (pages 24, 48, 51, 52, 117, 146, 147).

Amgoud, Leila, Yannis Dimopoulos, and Pavlos Moraitis (2008b). “Making Decisions through Preference-Based Argumentation”. In: *Knowledge Representation*. Ed. by Gerhard Brewka and Jérôme Lang. AAAI Press, pp. 113–123 (page 23).

Amgoud, Leila, Claudette Cayrol, Marie-Christine Lagasquie-Schiex, and P. Livet (2008c). “On bipolarity in argumentation frameworks”. In: *International Journal of Intelligent Systems* 23.10, pp. 1062–1093 (page 33).

Artikis, Alexander, Marek J. Sergot, and Jeremy V. Pitt (2009). “Specifying norm-governed computational societies”. In: *ACM Trans. Comput. Log.* 10.1 (page 77).

Artikis, Alexander, Robert Craven, Nihan Kesim Cicekli, Babak Sadighi, and Kostas Stathis, eds. (2012). *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*. Vol. 7360. Lecture Notes in Computer Science. Springer.

Atkinson, Katie Marie (2005). “What should we do?: Computational representation of persuasive argument in practical reasoning”. PhD thesis (pages 22, 25, 37, 40, 41, 43, 49, 106, 107).

Atkinson, Katie and Trevor J. M. Bench-Capon (2007a). “Action-Based Alternating Transition Systems for Arguments about Action”. In: *AAAI*. AAAI Press, pp. 24–29 (page 37).

— (Nov. 5, 2007b). “Practical reasoning as presumptive argumentation using action based alternating transition systems.” In: *Artificial Intelligence* 171.10-15, pp. 855–874 (pages 15, 37, 40, 42, 47, 49, 51, 52, 106, 146, 147).

Atkinson, Katie and Trevor J. M. Bench-Capon (2014a). “States, goals and values: Revisiting practical reasoning”. In: *ArgMAS* (pages 49, 51).

— (2014b). “Taking the Long View: Looking Ahead in Practical Reasoning”. In: *Computational Models of Argument - Proceedings of COMMA 2014, Atholl Palace Hotel, Scottish Highlands, UK, September 9-12, 2014*. Ed. by Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti. Vol. 266. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 109–120 (pages 49, 51).

Atkinson, Katie, Trevor J. M. Bench-Capon, and Peter McBurney (2004). “PARMENIDES: Facilitating Democratic Debate”. In: *Electronic Government: Third International Conference, EGOV 2004, Zaragoza, Spain, August 30 - September 3, 2004, Proceedings*. Ed. by Roland Traunmüller. Vol. 3183. Lecture Notes in Computer Science. Springer, pp. 313–316 (page 152).

Atkinson, Katie, Trevor J. M. Bench-Capon, and Sanjay Modgil (2006). “Argumentation for Decision Support”. In: *Database and Expert Systems Applications, 17th International Conference, DEXA 2006, Kraków, Poland, September 4-8, 2006, Proceedings*. Ed. by Stéphane

Bressan, Josef Küng, and Roland Wagner. Vol. 4080. Lecture Notes in Computer Science. Springer, pp. 822–831 (page 152).

Atkinson, Katie, Trevor J. M. Bench-Capon, Dan Cartwright, and Adam Zachary Wyner (2011). "Semantic models for policy deliberation". In: *The 13th International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 6-10, 2011, Pittsburgh, PA, USA*. Ed. by Kevin D. Ashley and Tom M. van Engers. ACM, pp. 81–90 (page 37).

Axelrod, Robert M., ed. (1976). *Structure of decision: The cognitive maps of political elites*. Princeton, NJ: Princeton University Press (page 152).

Balke, Tina, Marina De Vos, Julian A. Padget, and Dimitris Traskas (2011). "On-line reasoning for institutionally-situated BDI agents". In: *Autonomous Agents and Multiagent Systems (AAMAS)*. Ed. by Liz Sonenberg, Peter Stone, Kagan Tumer, and Pinar Yolum. IFAAMAS, pp. 1109–1110 (page 76).

Balke, Tina, Marina De Vos, and Julian A. Padget (2013). "Evaluating the Cost of Enforcement by Agent-Based Simulation: A Wireless Mobile Grid Example". In: *PRIMA 2013: Principles and Practice of Multi-Agent Systems - 16th International Conference, Dunedin, New Zealand, December 1-6, 2013. Proceedings*. Ed. by Guido Boella, Edith Elkind, Bastin Tony Roy Savarimuthu, Frank Dignum, and Martin K. Purvis. Vol. 8291. Lecture Notes in Computer Science. Springer, pp. 21–36 (page 151).

Baral, Chitta (2003). *Knowledge Representation, Reasoning, and Declarative Problem Solving*. New York, NY, USA: Cambridge University Press. ISBN: 0521818028 (pages 17, 76, 78).

Barber, Salvador, Walter Bossert, and Prasanta K. Pattanaik (2001). *Ranking Sets of Objects*. Cahiers de recherche. Universite de Montreal, Departement de sciences economiques (pages 118, 129).

Barbini, Patrizio, Yining Wu, and Martin Caminada (2009). "An implementation of argument based discussion". In: *8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, 2009, Volume 2*. Ed. by Carles Sierra, Cristiano Castelfranchi, Keith S. Decker, and Jaime Simão Sichman. IFAAMAS, pp. 1425–1426 (page 46).

Baroni, Pietro and Massimiliano Giacomin (2003). "Solving Semantic Problems with Odd-Length Cycles in Argumentation". In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 7th European Conference, ECSQARU 2003, Aalborg, Denmark, July 2-*

5, 2003. Proceedings. Ed. by Thomas D. Nielsen and Nevin Lianwen Zhang. Vol. 2711. Lecture Notes in Computer Science. Springer, pp. 440–451 (page 116).

— (2009). "Argumentation in Artificial Intelligence". In: ed. by Iyad Rahwan. Springer. Chap. Semantics of Abstract Argument Systems, pp. 25–44 (pages 15, 45).

Belesiotis, Alexandros, Michael Rovatsos, and Iyad Rahwan (2009). "A Generative Dialogue System for Arguing about Plans in Situation Calculus". In: *Argumentation in Multi-Agent Systems, 6th International Workshop, ArgMAS 2009, Budapest, Hungary, May 12, 2009. Revised Selected and Invited Papers*. Ed. by Peter McBurney, Iyad Rahwan, Simon Parsons, and Nicolas Maudet. Vol. 6057. Lecture Notes in Computer Science. Springer, pp. 23–41 (page 145).

Bench-Capon, Trevor J. M. (2002). "Value-based argumentation frameworks". In: *Non Monotonic Reasoning*. Ed. by Salem Benferhat and Enrico Giunchiglia, pp. 443–454 (pages 35, 103).

— (2003). "Persuasion in Practical Argument Using Value-based Argumentation Frameworks". In: *Logic Computant* 13.3, pp. 429–448 (page 102).

Bench-Capon, Trevor J. M. and Katie Atkinson (2009). "Action-State Semantics for Practical Reasoning". In: *The Uses of Computational Argumentation, Papers from the 2009 AAAI Fall Symposium, Arlington, Virginia, USA, November 5-7, 2009*. Vol. FS-09-06. AAAI Technical Report. AAAI (pages 52, 146).

Bench-Capon, Trevor J. M. and Paul E. Dunne (2007). "Argumentation in artificial intelligence". In: *Artificial Intelligence* 171.10-15, pp. 619–641 (page 144).

Bench-Capon, Trevor J. M. and Geof Staniford (1995). "PLAID: Proactive Legal Assistance". In: *Proceedings of the Fifth International Conference on Artificial Intelligence and Law, ICAIL '95, College Park, Maryland, USA, May 21-24, 1995*. Ed. by L. Thorne McCarty. ACM, pp. 81–88 (page 37).

Bench-Capon, Trevor J. M., Katie Atkinson, and Peter McBurney (2012). "Using argumentation to model agent decision making in economic experiments". In: *Autonomous Agents and Multi-Agent Systems* 25.1, pp. 183–208 (page 152).

Bench-Capon, Trevor, Henry Prakken, and Giovanni Sartor (2009). "Argumentation in Artificial Intelligence". In: ed. by Iyad Rahwan and G. Simari. Springer. Chap. Argumentation in Legal Reasoning, pp. 363–382 (page 15).

Bentahar, Jamal, Bernard Moulin, and Brahim Chaib-draa (2004). “Specifying and Implementing a Persuasion Dialogue Game Using Commitments and Arguments”. In: *Argumentation in Multi-Agent Systems, First International Workshop, ArgMAS 2004, New York, NY, USA, July 19, 2004, Revised Selected and Invited Papers*. Ed. by Iyad Rahwan, Pavlos Moraitis, and Chris Reed. Vol. 3366. Lecture Notes in Computer Science. Springer, pp. 130–148 (pages 32, 43).

Besnard, Philippe and Anthony Hunter (2008). *Elements of Argumentation*. MIT Press (page 144).

Black, Elizabeth and Katie Atkinson (2009). “Dialogues that account for different perspectives in collaborative argumentation”. In: *8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, 2009, Volume 2*. Ed. by Carles Sierra, Cristiano Castelfranchi, Keith S. Decker, and Jaime Simão Sichman. IFAAMAS, pp. 867–874 (page 151).

Black, Elizabeth and Katie Bentley (2011). “An Empirical Study of a Deliberation Dialogue System”. In: *Theorie and Applications of Formal Argumentation - First International Workshop, TAFA 2011. Barcelona, Spain, July 16-17, 2011, Revised Selected Papers*. Ed. by Sanjay Modgil, Nir Oren, and Francesca Toni. Vol. 7132. Lecture Notes in Computer Science. Springer, pp. 132–146 (page 151).

Blount, Justin and Michael Gelfond (2012). “Reasoning about the Intentions of Agents”. In: *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*. Ed. by Alexander Artikis, Robert Craven, Nihan Kesim Cicekli, Babak Sadighi, and Kostas Stathis. Vol. 7360. Lecture Notes in Computer Science. Springer, pp. 147–171 (page 76).

Blum, Avrim L. and Merrick L. Furst (1997). “Fast planning through planning graph analysis”. In: *Artificial Intelligence* 90.1, pp. 281 –300 (pages 57, 58, 82).

Bodanza, Gustavo Adrian and Fernando A. Tohmé (2009). “Two approaches to the problems of self-attacking arguments and general odd-length cycles of attack”. In: *J. Applied Logic* 7.4, pp. 403–420 (page 116).

Boella, Guido and Leendert W. N. van der Torre (2004). “Regulative and Constitutive Norms in Normative Multiagent Systems”. In: *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada, June 2-5, 2004*, pp. 255–266 (page 27).

— (2008). “Substantive and procedural norms in normative multiagent systems”. In: *Journal of Applied Logic* 6.2, pp. 152–171 (page 27).

Boella, Guido, Leendert W. N. van der Torre, and Harko Verhagen (2006). “Introduction to normative multiagent systems”. In: *Computational & Mathematical Organization Theory* 12.2-3, pp. 71–79 (page 14).

Boer, Frank S. de, Koen V. Hindriks, Wiebe van der Hoek, and John-Jules Ch. Meyer (2002). “Agent Programming with Declarative Goals”. In: *CoRR* cs.AI/0207008 (pages 60, 67).

— (2007). “A verification framework for agent programming with declarative goals”. In: *Journal of Applied Logic* 5.2, pp. 277–302 (pages 67, 148).

Bondarenko, Andrei, Francesca Toni, and Robert A. Kowalski (1993). “An Assumption-Based Framework for Non-Monotonic Reasoning”. In: *LPNMR*, pp. 171–189 (pages 35, 36).

Bonet, Blai and Hector Geffner (1996). “Arguing for Decisions: A Qualitative Model of Decision Making”. In: *UAI ’96: Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence, Reed College, Portland, Oregon, USA, August 1-4, 1996*. Ed. by Eric Horvitz and Finn Verner Jensen. Morgan Kaufmann, pp. 98–105 (page 100).

Börger, Egon and Robert Stärk (2003). “Asynchronous Multi-Agent ASMs”. In: *Abstract State Machines*. Springer Berlin Heidelberg, pp. 207–282 (page 55).

Bratman, M. E. (1990). “What is intention?” In: *Intentions in communiation*. Ed. by Philip R Cohen; Jerry L Morgan; Martha E Pollack. The MIT Press: Cambridge, MA (page 23).

Bratman, Michael E. (1987). *Intention, Plans, and Practical Reason*. Harvard University Press (page 20).

Brewka, Gerhard and Thomas Eiter (2000). “Prioritizing Default Logic”. In: *Intellectics and Computational Logic (to Wolfgang Bibel on the occasion of his 60th birthday)*, pp. 27–45 (page 112).

Broersen, J., M. Dastani, J. Hulstijn, and L. van der Torre (2002). “Goal generation in the BOID architecture”. In: *Cognitive Science Quarterly* 2.3-4, pp. 428–447 (pages 66, 67, 114).

Broersen, Jan, Mehdi Dastani, Joris Hulstijn, Zisheng Huang, and Leendert van der Torre (2001). “The BOID Architecture: Conflicts Between Beliefs, Obligations, Intentions and Desires”. In: *Proceedings of the Fifth International Conference on Autonomous Agents*.

AGENTS ’01. Montreal, Quebec, Canada: ACM, pp. 9–16 (pages 15, 29, 47, 51, 132, 145).

Caminada, M.W.A. and M. Podlaszewski (2012a). “User-Computer Persuasion Dialogue for Grounded Semantics”. In: *BENELUX CONFERENCE ON ARTIFICIAL INTELLIGENCE (BNIAC), Maastricht*, pp. 343–344 (page 151).

Caminada, Martin (2004). “Dialogues and HY-arguments”. In: *10th International Workshop on Non-Monotonic Reasoning (NMR 2004), Whistler, Canada, June 6-8, 2004, Proceedings*. Ed. by James P. Delgrande and Torsten Schaub, pp. 94–99 (page 134).

— (2006). “On the Issue of Reinstatement in Argumentation”. In: *Logics in Artificial Intelligence, 10th European Conference, JELIA 2006, Liverpool, UK, September 13-15, 2006, Proceedings*. Ed. by Michael Fisher, Wiebe van der Hoek, Boris Konev, and Alexei Lisitsa. Vol. 4160. Lecture Notes in Computer Science. Springer, pp. 111–123 (pages 34, 46, 114, 115, 135).

— (2008). “A formal account of Socratic-style argumentation”. In: *J. Applied Logic* 6.1, pp. 109–132 (page 45).

— (2010). “Preferred Semantics as Socratic Discussion”. In: *Proceedings of the eleventh AI*IA symposium on artificial intelligence*. Ed. by Alfonso E. Gerevini and Alessandro Saetti, pp. 209–216 (pages 46, 113, 135).

Caminada, Martin and Leila Amgoud (2007). “On the evaluation of argumentation formalisms”. In: *Artif. Intell.* 171.5-6, pp. 286–310 (pages 104, 129, 141, 142).

Caminada, Martin and Dov M. Gabbay (2009). “A Logical Account of Formal Argumentation”. In: *Studia Logica* 93.2-3, pp. 109–145 (pages 116, 136).

Caminada, Martin and Mikolaj Podlaszewski (2012b). “Grounded Semantics as Persuasion Dialogue”. In: *Computational Models of Argument - Proceedings of COMMA 2012, Vienna, Austria, September 10-12, 2012*. Ed. by Bart Verheij, Stefan Szeider, and Stefan Woltran. Vol. 245. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 478–485 (pages 32, 43, 45, 46, 133, 134).

Caminada, Martin, Sanjay Modgil, and Nir Oren (2014a). “Preferences and Unrestricted Rebut”. In: *Computational Models of Argument - Proceedings of COMMA 2014, Atholl Palace Hotel, Scottish Highlands, UK, September 9-12, 2014*. Ed. by Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti. Vol. 266. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 209–220 (pages 102, 103, 118).

Caminada, Martin, Wolfgang Dvork, and Srdjan Vesic (2014b). "Preferred Semantics as Socratic Discussion". In: *Journal of Logic and Computation(JLC)* (pages 17, 113, 133, 135, 136, 141).

Caminada, Martin, Roman Kutlak, Nir Oren, and Wamberto Weber Vasconcelos (2014c). "Scrutable plan enactment via argumentation and natural language generation". In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*. Ed. by Ana L. C. Bazzan, Michael N. Huhns, Alessio Lomuscio, and Paul Scerri. IFAAMAS/ACM, pp. 1625–1626 (pages 15, 20, 45, 46, 133, 144, 145, 151).

Cayrol, Claudette, Sylvie Doutre, and Jérôme Mengin (2001). "Dialectical Proof Theories for the Credulous Preferred Semantics of Argumentation Frameworks". In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 6th European Conference, ECSQARU 2001, Toulouse, France, September 19-21, 2001, Proceedings*. Ed. by Salem Benferhat and Philippe Besnard. Vol. 2143. Lecture Notes in Computer Science. Springer, pp. 668–679 (pages 45, 46, 135, 141).

Chesani, Federico, Paola Mello, Marco Montali, and Paolo Torroni (2013). "Representing and monitoring social commitments using the event calculus". In: *Autonomous Agents and Multi-Agent Systems* 27.1, pp. 85–130 (page 62).

Chesñevar, Carlos Iván, Guillermo Ricardo Simari, Teresa Alsinet, and Lluis Godo (2012). "A Logic Programming Framework for Possibilistic Argumentation with Vague Knowledge". In: *CoRR* abs/1207.4123 (page 144).

Cliffe, Owen (2007). "Specifying and Analysing Institutions in Multi-Agent Systems Using Answer Set Programming". PhD thesis. Department of Computer Science, University of Bath (pages 76, 80, 150).

Cliffe, Owen, Marina De Vos, and Julian A. Padget (2005). "Specifying and Analysing Agent-Based Social Institutions Using Answer Set Programming". In: *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems, AAMAS 2005 International Workshops on Agents, Norms and Institutions for Regulated Multi-Agent Systems, ANIREM 2005, and Organizations in Multi-Agent Systems, OOOP 2005, Utrecht, The Netherlands, July 25-26, 2005, Revised Selected Papers*. Ed. by Olivier Boissier, Julian A. Padget, Virginia Dignum, Gabriela Lindemann, Eric T. Matson, Sascha Ossowski, Jaime Simão

Sichman, and Javier Vázquez-Salceda. Vol. 3913. Lecture Notes in Computer Science. Springer, pp. 99–113 (page 61).

Colmerauer, Alain and Philippe Roussel (1993). “The Birth of Prolog”. In: *History of Programming Languages Conference (HOPL-II), Preprints, Cambridge, Massachusetts, USA, April 20-23, 1993*. Ed. by John A. N. Lee and Jean E. Sammet. ACM, pp. 37–52 (page 76).

Conte, Rosaria, Rino Falcone, and Giovanni Sartor (1999). “Introduction: Agents and Norms: How to Fill the Gap?” In: *Artificial Intelligence Law* 7.1, pp. 1–15 (page 26).

Criado, Natalia, Estefania Argente, and Vicente J. Botti (2010). “A BDI architecture for normative decision making”. In: *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*. Ed. by Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen. IFAAMAS, pp. 1383–1384 (pages 15, 47, 132, 145).

Criado, Natalia, Elizabeth Black, and Michael Mordechai Luck (2015). “A Coherence Maximisation Process For Solving Normative Inconsistencies”. In: *Autonomous Agents and Multi-Agent Systems*. ISSN: 1387-2532 (pages 69, 70, 146).

Dastani, Mehdi, Joris Hulstijn, and Leendert W. N. van der Torre (2005). “How to decide what to do?” In: *European Journal of Operational Research* 160.3, pp. 762–784 (page 24).

Dastani, Mehdi, Davide Grossi, John-Jules Ch. Meyer, and Nick A. M. Tinnemeier (2009). “Normative Multi-Agent Programs and Their Logics”. In: *Normative Multi-Agent Systems, 15.03. - 20.03.2009*. Ed. by Guido Boella, Pablo Noriega, Gabriella Pigozzi, and Harko Verhagen. Vol. 09121. Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany (pages 28, 61).

De Vos, Marina, Tina Balke, and Ken Satoh (2013). “Combining event-and state-based norms”. In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS ’13, Saint Paul, MN, USA, May 6-10, 2013*. Ed. by Maria L. Gini, Onn Shehory, Takayuki Ito, and Catholijn M. Jonker. IFAAMAS, pp. 1157–1158 (pages 61, 149).

Delgrande, James P. and Torsten Schaub, eds. (2004). *10th International Workshop on Non-Monotonic Reasoning (NMR 2004), Whistler, Canada, June 6-8, 2004, Proceedings*.

Delgrande, James P., Torsten Schaub, Hans Tompits, and Kewen Wang (2004). “A Classification and Survey of Preference Handling Approaches in Nonmonotonic Reasoning”. In: *Computational Intelligence* 20.2, pp. 308–334 (page 105).

Devereux, Joseph and Chris Reed (2009). “Strategic Argumentation in Rigorous Persuasion Dialogue”. In: *Argumentation in Multi-Agent Systems, 6th International Workshop, ArgMAS 2009, Budapest, Hungary, May 12, 2009. Revised Selected and Invited Papers*. Ed. by Peter McBurney, Iyad Rahwan, Simon Parsons, and Nicolas Maudet. Vol. 6057. Lecture Notes in Computer Science. Springer, pp. 94–113 (page 43).

Dignum, Virginia (2004). “A Model for Organizational Interaction: Based on Agents, Founded in Logics”. PhD thesis. Utrecht University (page 61).

Dimopoulos, Yannis and Alberto Torres (1996). “Graph Theoretical Structures in Logic Programs and Default Theories”. In: *Theor. Comput. Sci.* 170.1-2, pp. 209–244 (page 134).

Doherty, Patrick, Joakim Gustafsson, Lars Karlsson, and Jonas Kvarnström (1998). “TAL: Temporal Action Logics Language Specification and Tutorial”. In: *Electron. Trans. Artif. Intell.* 2, pp. 273–306 (pages 54, 77).

Doyle, Jon and Richmond H. Thomason (1999). “Background to Qualitative Decision Theory”. In: *AI Magazine* 20.2, pp. 55–68 (pages 25, 111).

Dung, Phan Minh (1995). “On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games”. In: *Artificial Intelligence* 77.2, pp. 321–358 (pages 15, 32–34, 46, 100, 102, 105, 113, 134).

Dung, Phan Minh, Robert A. Kowalski, and F. Toni (2009). “Assumption-Based Argumentation”. In: *Argumentation in Artificial Intelligence*. Springer (pages 35, 36, 39, 46, 129, 133).

Dunne, Paul E. and Trevor J. M. Bench-Capon (2004). “Complexity in Value-Based Argument Systems”. In: *JELIA*. Ed. by José Júlio Alferes and João Alexandre Leite. Vol. 3229. Lecture Notes in Computer Science. Springer, pp. 360–371 (pages 35, 36).

Eemeren, Frans H. Van, Rob Grootendorst, and Francisca Snoeck Henkemans (1996). *Fundamentals of argumentation theory: A handbook of historical backgrounds and contemporary developments*. Lawrence Erlbaum Associates (page 32).

Eiter, Thomas, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer (1999). “The Diagnosis Frontend of the dlv System”. In: *AI Commun.* 12.1-2, pp. 99–111 (page 76).

Eiter, Thomas, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres (2011). “Answer Set Planning Under Action Costs”. In: *CoRR* abs/1106.5257 (pages 76, 97).

Erdem, Esra, Doga Gizem Kisa, Umut Öztok, and Peter Schüller (2013). "A General Formal Framework for Pathfinding Problems with Multiple Agents". In: *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. Ed. by Marie desJardins and Michael L. Littman. AAAI Press (page 97).

Esteva, Marc, Juan A. Rodríguez-Aguilar, Carles Sierra, Pere Garcia, and Josep Lluís Arcos (2001). "On the Formal Specifications of Electronic Institutions". In: *Agent Mediated Electronic Commerce, The European AgentLink Perspective.* Ed. by Frank Dignum and Carles Sierra. Vol. 1991. Lecture Notes in Computer Science. Springer, pp. 126–147 (pages 13, 28).

Fan, Xiuyi and Francesca Toni (2012). "Agent Strategies for ABA-based Information-seeking and Inquiry Dialogues". In: *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*. Ed. by Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas. Vol. 242. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 324–329 (page 44).

— (2015). "On Computing Explanations in Argumentation". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. Ed. by Blai Bonet and Sven Koenig. AAAI Press, pp. 1496–1502 (pages 15, 20, 45, 134, 144, 145).

Fan, Xiuyi, Robert Craven, Ramsay Singer, Francesca Toni, and Matthew Williams (2013). "Assumption-Based Argumentation for Decision-Making with Preferences: A Medical Case Study". In: *Computational Logic in Multi-Agent Systems - 14th International Workshop, CLIMA XIV, Corunna, Spain, September 16-18, 2013. Proceedings*. Ed. by João Leite, Tran Cao Son, Paolo Torroni, Leon van der Torre, and Stefan Woltran. Vol. 8143. Lecture Notes in Computer Science. Springer, pp. 374–390. ISBN: 978-3-642-40623-2 (page 129).

Ferrando, Sergio Pajares, Eva Onaindia, and Alejandro Torreño (2011). "An Architecture for Defeasible-Reasoning-Based Cooperative Distributed Planning". In: *On the Move to Meaningful Internet Systems: OTM 2011 - Confederated International Conferences: CoopIS, DOA-SVI, and ODBASE 2011, Hersonissos, Crete, Greece, October 17-21, 2011, Proceedings, Part I*. Ed. by Robert Meersman, Tharam S. Dillon, Pilar Herrero, Akhil Kumar, Manfred Reichert, Li Qing, Beng Chin Ooi, Ernesto Damiani, Douglas C. Schmidt, Jules

White, Manfred Hauswirth, Pascal Hitzler, and Mukesh K. Mohania. Vol. 7044. Lecture Notes in Computer Science. Springer, pp. 200–217 (pages 144, 151).

Fikes, Richard E. and Nils J. Nilsson (1971). “STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving”. In: *Proceedings of the 2Nd International Joint Conference on Artificial Intelligence*. IJCAI’71. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 608–620 (pages 54, 56, 77, 99).

Fornara, Nicoletta and Marco Colombetti (2007). “Specifying and Enforcing Norms in Artificial Institutions”. In: *Normative Multi-agent Systems, 18.03. - 23.03.2007*. Ed. by Guido Boella, Leendert W. N. van der Torre, and Harko Verhagen. Vol. 07122. Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (page 61).

Fox, John and Simon Parsons (1998). “Arguing about beliefs and actions”. In: *Applications of Uncertainty Formalisms*. Ed. by Anthony Hunter and Simon Parsons. Vol. 1455. Lecture Notes in Computer Science. Springer, pp. 266–302 (pages 21, 25, 112).

Fox, Maria and Derek Long (2003). “PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains”. In: *Journal of Artificial Intelligence Research (JAIR)* 20, pp. 61–124 (pages 54, 57).

Gaertner, Dorian (2008). “Argumentation and Normative Reasoning”. PhD thesis. University of London, Imperial College London (pages 44, 48, 51).

Gaertner, Dorian and Francesca Toni (2007a). “CaSAPI: a system for credulous and sceptical argumentation”. In: *First International Workshop on Argumentation and Non-monotonic Reasoning*, pp. 80–95 (pages 36, 48).

— (2007b). “Preferences and Assumption-Based Argumentation for Conflict-Free Normative Agents”. In: *Argumentation in Multi-Agent Systems, 4th International Workshop (ArgMAS)*, pp. 94–113 (page 15).

García-Camino, Andrés, Pablo Noriega, and Juan A. Rodríguez-Aguilar (2005). “Implementing norms in electronic institutions”. In: *4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands*. Ed. by Frank Dignum, Virginia Dignum, Sven Koenig, Sarit Kraus, Munindar P. Singh, and Michael Wooldridge. ACM, pp. 667–673 (pages 28, 61).

García, Alejandro Javier, Carlos Iván Chesñevar, Nicolás D. Rotstein, and Guillermo Ricardo Simari (2013). “Formalizing dialectical explanation support for argument-based reasoning in knowledge-based systems”. In: *Expert Syst. Appl.* 40.8, pp. 3233–3247 (pages 15, 45, 144).

Garrido, Antonio, Maria Fox, and Derek Long (2002). “A Temporal Planning System for Durative Actions of PDDL2.1”. In: *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI’2002, Lyon, France, July 2002*. Ed. by Frank van Harmelen. IOS Press, pp. 586–590 (page 57).

Gasparini, Luca, Timothy J. Norman, Martin J. Kollingbaum, Liang Chen, and John-Jules Ch. Meyer (2015). “Verifying Normative System Specification containing Collective Imperatives and Deadlines”. In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*. Ed. by Gerhard Weiss, Pinar Yolum, Rafael H. Bordini, and Edith Elkind. ACM, pp. 1821–1822 (page 62).

Gasque, A. Rolando Medellin (2013). “Argumentation-based Dialogues over Cooperative Plans”. PhD thesis. University of Liverpool (pages 37, 44, 52, 106, 143, 145, 147, 151).

Gebser, Martin, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius Thomas Schneider (2011). “Potassco: The Potsdam Answer Set Solving Collection”. In: *AI Commun.* 24.2, pp. 107–124 (page 76).

Gebser, Martin, Roland Kaminski, Philipp Obermeier, and Torsten Schaub (2015). “Ricochet Robots Reloaded: A Case-Study in Multi-shot ASP Solving”. In: *Advances in Knowledge Representation, Logic Programming, and Abstract Argumentation - Essays Dedicated to Gerhard Brewka on the Occasion of His 60th Birthday*. Ed. by Thomas Eiter, Hannes Strass, Miroslaw Truszczynski, and Stefan Woltran. Vol. 9060. Lecture Notes in Computer Science. Springer, pp. 17–32 (page 149).

Gelfond, Michael (2004). “Answer Set Programming and the Design of Deliberative Agents”. In: *Logic Programming, 20th International Conference, ICLP 2004, Saint-Malo, France, September 6-10, 2004, Proceedings*. Ed. by Bart Demoen and Vladimir Lifschitz. Vol. 3132. Lecture Notes in Computer Science. Springer, pp. 19–26 (page 76).

Gelfond, Michael and Vladimir Lifschitz (1988). “The Stable Model Semantics for Logic Programming”. In: *ICLP/SLP*. Ed. by Robert A. Kowalski and Kenneth A. Bowen. MIT Press, pp. 1070–1080 (pages 76, 79).

— (1991). "Classical Negation in Logic Programs and Disjunctive Databases". In: *New Generation Computing* 9.3/4, pp. 365–386 (page 76).

— (1998). "Action Languages". In: *Electronic Transactions on AI* 3 (page 77).

Giannikis, Georgios K. and Aspassia Daskalopulu (2011). "Normative conflicts in electronic contracts". In: *Electronic Commerce Research and Applications* 10.2, pp. 247–267 (page 69).

Gini, Maria L., Onn Shehory, Takayuki Ito, and Catholijn M. Jonker, eds. (2013). *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013*. IFAAMAS.

Glasspool, David, John Fox, Ayelet Oettinger, and James Smith-Spark (2006). "Argumentation in Decision Support for Medical Care Planning for Patients and Clinicians". In: *Argumentation for Consumers of Healthcare, Papers from the 2006 AAAI Spring Symposium, Technical Report SS-06-01, Stanford, California, USA, March 27-29, 2006*. AAAI, pp. 58–63 (page 152).

Gottlob, Georg and Toby Walsh, eds. (2003). *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Morgan Kaufmann.

Governatori, Guido, Francesco Olivieri, Simone Scannapieco, and Matteo Cristani (2011). "Designing for Compliance: Norms and Goals". In: *Rule-Based Modeling and Computing on the Semantic Web, 5th International Symposium, RuleML 2011- America, Ft. Lauderdale, FL, Florida, USA, November 3-5, 2011. Proceedings*. Ed. by Frank Olken, Monica Palmirani, and Davide Sottara. Vol. 7018. Lecture Notes in Computer Science. Springer, pp. 282–297 (page 151).

Gu, Jun, Paul W. Purdom, John Franco, and Benjamin W. Wah (1997). "Algorithms for the Satisfiability (SAT) Problem: A Survey". In: *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, pp. 19–152 (page 77).

Hamblin, C. L. (1970). *Fallacies*. Methuen & Co. Ltd (pages 32, 41).

— (1971). "Mathematical models of dialogues". In: *Theoria* 37, pp. 130–155 (pages 32, 41).

Harmelen, Frank van, ed. (2002). *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI'2002, Lyon, France, July 2002*. IOS Press.

Hindriks, Koen V. and M. Birna van Riemsdijk (2007). “Satisfying Maintenance Goals”. In: *Declarative Agent Languages and Technologies V, 5th International Workshop, DALT 2007, Honolulu, HI, USA, May 14, 2007, Revised Selected and Invited Papers*. Ed. by Matteo Baldoni, Tran Cao Son, M. Birna van Riemsdijk, and Michael Winikoff. Vol. 4897. Lecture Notes in Computer Science. Springer, pp. 86–103 (page 148).

Hindriks, Koen V., Wiebe van der Hoek, and M. Birna van Riemsdijk (2009). “Agent programming with temporally extended goals”. In: *8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, 2009, Volume 1*. Ed. by Carles Sierra, Cristiano Castelfranchi, Keith S. Decker, and Jaime Simão Sichman. IFAAMAS, pp. 137–144 (page 148).

Hoek, Wiebe van der, Mark Roberts, and Michael Wooldridge (2007). “Social laws in alternating time: effectiveness, feasibility, and synthesis”. In: *Synthese* 156.1, pp. 1–19 (pages 47, 49).

Hübner, Jomi Fred, Jaime Simão Sichman, and Olivier Boissier (2007). “Developing organised multiagent systems using the MOISE”. In: *IJAOSE* 1.3/4, pp. 370–395 (page 61).

Hulstijn, Joris and Leendert W. N. van der Torre (2004). “Combining goal generation and planning in an argumentation framework”. In: *Non Monotonic Reasoning*. Ed. by James P. Delgrande and Torsten Schaub, pp. 212–218 (pages 15, 24, 47, 48, 51, 52, 66, 118, 144, 146, 147).

Introne, Joshua and Luca Iandoli (2014). “Improving decision-making performance through argumentation: An argument-based decision support system to compute with evidence”. In: *Decision Support Systems* 64, pp. 79–89 (page 152).

Jakobovits, H. and D. Vermeir (1999). “Dialectic semantics for argumentation frameworks”. In: *Seventh International Conference on Artificial Intelligence and Law, New York*, pp. 53–62 (page 134).

Jeffrey, Richard (1983). *The Logic of Decision*. Vol. 77. 2. University of Chicago Press, pp. 250–253 (page 22).

Jennings, Nick R. (1993). “Commitments and conventions: The foundation of coordination in multi-agent systems”. In: *Knowledge Engineering Review* 8.3, pp. 223–250 (page 26).

Jonker, Geert, Frank Dignum, and John-Jules Ch. Meyer (2007). “Achieving cooperation among selfish agents in the air traffic management domain using signed money”. In: *6th Interna-*

tional Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007), Honolulu, Hawaii, USA, May 14-18, 2007. Ed. by Edmund H. Durfee, Makoto Yokoo, Michael N. Huhns, and Onn Shehory. IFAAMAS, p. 213 (page 151).

Kafali, Özgür, Akin Günay, and Pinar Yolum (2014). "GOSU: computing GOal SUpport with commitments in multiagent systems". In: *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*. Ed. by Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan. Vol. 263. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 477–482 (page 62).

Kakas, Antonis C. and Pavlos Moraitis (2003). "Argumentation based decision making for autonomous agents". In: *The Second International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2003, July 14-18, 2003, Melbourne, Victoria, Australia, Proceedings*. ACM, pp. 883–890 (pages 20, 132).

Kakas, Antonis C., Paolo Mancarella, Fariba Sadri, Kostas Stathis, and Francesca Toni (2004). "The KGP Model of Agency". In: *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*. Ed. by Ramon López de Mántaras and Lorenza Saitta. IOS Press, pp. 33–37. ISBN: 1-58603-452-9 (page 30).

Knoblock, Craig A. (1994). "Generating Parallel Execution Plans with a Partial-order Planner". In: *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems, University of Chicago, Chicago, Illinois, USA, June 13-15, 1994*. Ed. by Kristian J. Hammond. AAAI, pp. 98–103 (page 57).

Kok, Eric M., John-Jules Ch. Meyer, Henry Prakken, and Gerard Vreeswijk (2012). "Testing the benfits of structured argumentation in multi-agent deliberation dialogues". In: *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*. Ed. by Wiebe van der Hoek, Lin Padgham, Vincent Conitzer, and Michael Winikoff. IFAAMAS, pp. 1411–1412 (page 44).

Kollingbaum, Martin J. and Timothy J. Norman (2003). "NoA - A Normative Agent Architecture". In: *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, pp. 1465–1466 (pages 13, 15, 47, 132, 146).

Kollingbaum, Martin (2005). "Norm-governed Practical Reasonig Agents". PhD thesis. University of Aberdeen (pages 27, 30, 51).

Kowalski, R and M Sergot (Jan. 1986). “A Logic-based Calculus of Events”. In: *New Gen. Comput.* 4.1, pp. 67–95 (pages 54, 77).

Kraus, Sarit, Katia P. Sycara, and Amir Evenchik (1998). “Reaching Agreements Through Argumentation: A Logical Model and Implementation”. In: *Artif. Intell.* 104.1-2, pp. 1–69 (pages 32, 36, 44).

Lacave, Carmen and Francisco Javier Díez (2004). “A review of explanation methods for heuristic expert systems”. In: *Knowledge Eng. Review* 19.2, pp. 133–146 (pages 14, 15, 45).

Lawrence, Michael, Paul Goodwin, and Robert Fildes (2002). “Influence of user participation on DSS use and decision accuracy”. In: *Omega* 30.5, pp. 381–392 (page 152).

Lee, Jeehang, Tingting Li, Marina De Vos, and Julian A. Padget (2013). “Governing intelligent virtual agent behaviour with norms”. In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS ’13, Saint Paul, MN, USA, May 6-10, 2013*. Ed. by Maria L. Gini, Onn Shehory, Takayuki Ito, and Catholijn M. Jonker. IFAAMAS, pp. 1205–1206 (page 150).

Lee, Joohyung and Ravi Palla (2010). “Situation Calculus as Answer Set Programming”. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. Ed. by Maria Fox and David Poole. AAAI Press (page 77).

— (2012). “Reformulating Temporal Action Logics in Answer Set Programming”. In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. Ed. by Jörg Hoffmann and Bart Selman. AAAI Press (pages 77, 99).

— (2014). “Reformulating the Situation Calculus and the Event Calculus in the General Theory of Stable Models and in Answer Set Programming”. In: *CoRR* abs/1401.4607 (pages 77, 99).

Lee, Joohyung, Vladimir Lifschitz, and Ravi Palla (2008). “A Reductive Semantics for Counting and Choice in Answer Set Programming”. In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. Ed. by Dieter Fox and Carla P. Gomes. AAAI Press, pp. 472–479 (page 80).

Li, Tingting (2014). "Normative Conflict Detection and Resolution in Cooperating Institutions". PhD thesis. University of Bath (pages 76, 150).

Li, Tingting, Tina Balke, Marina De Vos, Julian Padget, and Ken Satoh (2013). "Legal Conflict Detection in Interacting Legal Systems". In: *Legal Knowledge and Information Systems - JURIX 2013: The Twenty-Sixth Annual Conference, December 11-13, 2013, University of Bologna, Italy*. Ed. by Kevin D. Ashley. Vol. 259. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 107–116 (page 150).

Li, Zimi and Simon Parsons (2015). "On Argumentation with Purely Defeasible Rules". In: *Scalable Uncertainty Management - 9th International Conference, SUM 2015, Québec City, QC, Canada, September 16-18, 2015. Proceedings*. Ed. by Christoph Beierle and Alex Dekhtyar. Vol. 9310. Lecture Notes in Computer Science. Springer, pp. 330–343 (page 129).

Lifschitz, Vladimir (2002). "Answer set programming and plan generation". In: *Artif. Intell.* 138.1-2, pp. 39–54 (pages 76, 97).

Lin, Fangzhen and Yoav Shoham (1989). "Argument Systems: A Uniform Basis for Nonmonotonic Reasoning". In: *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR'89). Toronto, Canada, May 15-18 1989.* Ed. by Ronald J. Brachman, Hector J. Levesque, and Raymond Reiter. Morgan Kaufmann, pp. 245–255. ISBN: 1-55860-032-9 (page 33).

López, Fabiola López y and Michael Luck (2003). "Modelling Norms for Autonomous Agents". In: *4th Mexican International Conference on Computer Science (ENC)*. IEEE Computer Society, pp. 238–245 (page 27).

López, Fabiola López y, Michael Luck, and Mark d'Inverno (2005). "A Normative Framework for Agent-Based Systems". In: *Normative Multi-Agent Systems (NORMAS)*, pp. 24–35 (pages 13, 28, 30, 31, 51, 61, 67).

Mackenzie, Jim (1979). "Question-begging in non-cumulative systems". In: *Philosophical Logic* 8, pp. 117–133 (page 43).

— (1990). "Four Dialogue Systems". In: *Studia Logica* 49.4, pp. 567–583 (page 43).

Mark Law, Alessandra Russo and Krysia Broda (2015). *Simplified reduct for choice rules in ASP*. Tech. rep. Imperial College (page 80).

Marshall, C. C. (1989). “Representing the Structure of a Legal Argument”. In: *Proceedings of the 2Nd International Conference on Artificial Intelligence and Law*. ICAIL ’89. New York, NY, USA: ACM (page 37).

McBurney, Peter, Rogier M. van Eijk, Simon Parsons, and Leila Amgoud (2003). “A Dialogue Game Protocol for Agent Purchase Negotiations”. In: *Autonomous Agents and Multi-Agent Systems* 7.3, pp. 235–273 (page 44).

McBurney, Peter, Iyad Rahwan, Simon Parsons, and Nicolas Maudet, eds. (2010). *Argumentation in Multi-Agent Systems, 6th International Workshop, ArgMAS 2009, Budapest, Hungary, May 12, 2009. Revised Selected and Invited Papers*. Vol. 6057. Lecture Notes in Computer Science. Springer.

Mcdermott, D., M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins (1998). *PDDL - The Planning Domain Definition Language*. Tech. rep. TR-98-003. Yale Center for Computational Vision and Control, (page 54).

Mercier, H. and D Sperber (2011). “Argumentation: Its Adaptiveness and Efficacy”. In: *Behavioral and Brain Sciences* 34, pp. 94–111 (page 152).

Modgil, Sanjay (2007). “An Abstract Theory of Argumentation That Accommodates Defeasible Reasoning About Preferences”. In: *The 11th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*. Ed. by Khaled Mellouli. Vol. 4724. Lecture Notes in Computer Science. Springer, pp. 648–659 (pages 35, 36).

— (2009). “Reasoning about preferences in argumentation frameworks”. In: *Artificial Intelligence* 173.9-10, pp. 901–934 (page 102).

Modgil, Sanjay and Trevor J. M. Bench-Capon (2011). “Metalevel argumentation”. In: *J. Log. Comput.* 21.6, pp. 959–1003 (page 105).

Modgil, Sanjay and Martin Caminada (2009). “Proof theories and algorithms for abstract argumentation frameworks”. In: *Argumentation in artificial intelligence*. Springer, pp. 105–129 (pages 46, 134, 135, 141).

Modgil, Sanjay and Michael Luck (2008). “Argumentation Based Resolution of Conflicts between Desires and Normative Goals”. In: *Argumentation in Multi-Agent Systems (ArgMAS)*. Ed. by Iyad Rahwan and Pavlos Moraitis. Vol. 5384. Lecture Notes in Computer Science. Springer, pp. 19–36 (page 68).

Modgil, Sanjay and Henry Prakken (2011). “Revisiting Preferences and Argumentation”. In: *The 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1021–1026 (page 104).

— (2013). “A general account of argumentation with preferences”. In: *Artif. Intell.* 195, pp. 361–397 (pages 102, 103, 141).

Modgil, Sanjay, Nir Oren, and Francesca Toni, eds. (2012). *Theorie and Applications of Formal Argumentation - First International Workshop, TAFA 2011. Barcelona, Spain, July 16-17, 2011, Revised Selected Papers*. Vol. 7132. Lecture Notes in Computer Science. Springer.

Moore, J.D. and W.R. Swartout (1988). *Explanation in expert systems: a survey*. Explanation in expert systems: a survey no. 228. University of Southern California, Information Sciences Institute. URL: https://books.google.co.uk/books?id=Y4Q9mgEACAAJ (page 142).

Moses, Yoram and Moshe Tennenholtz (1995). “Artificial Social Systems”. In: *Computers and Artificial Intelligence* 14.6 (pages 13, 26).

Moulin, Bernard, Hengameh Irandoust, Micheline Bélanger, and G. Desbordes (2002). “Explanation and Argumentation Capabilities: Towards the Creation of More Persuasive Agents”. In: *Artif. Intell. Rev.* 17.3, pp. 169–222 (pages 14, 15, 45, 142).

Mueller, Erik T. (2004). “Event Calculus Reasoning Through Satisfiability”. In: *Journal of Logic and Computation (JLC)* 14.5, pp. 703–730 (page 77).

Nielsen, Thomas D. and Nevin Lianwen Zhang, eds. (2003). *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 7th European Conference, ECSQARU 2003, Aalborg, Denmark, July 2-5, 2003. Proceedings*. Vol. 2711. Lecture Notes in Computer Science. Springer.

Nigam, Vivek and João Leite (2006). “A Dynamic Logic Programming Based System for Agents with Declarative Goals”. In: *Declarative Agent Languages and Technologies IV, 4th International Workshop, DALT 2006, Hakodate, Japan, May 8, 2006, Selected, Revised and Invited Papers*. Ed. by Matteo Baldoni and Ulle Endriss. Vol. 4327. Lecture Notes in Computer Science. Springer, pp. 174–190 (pages 60, 66).

Nisbett, Richard E. and Lee Ross (1980). *Human Inference: Strategies and Shortcomings of Social Judgment*. Prentice-Hall (page 152).

Norman, Timothy J. and Derek Long (1995). “Alarms: An Implementation of Motivated Agency”. In: *Intelligent Agents II, Agent Theories, Architectures, and Languages, IJCAI ’95, Workshop (ATAL), Montreal, Canada, August 19-20, 1995, Proceedings*. Ed. by Michael Wooldridge, Jörg P. Müller, and Milind Tambe. Vol. 1037. Lecture Notes in Computer Science. Springer, pp. 219–234 (page 13).

Nunes, I., José Luiz Fiadeiro, and Wladyslaw M. Turski (1997). “Coordination Durative Actions”. In: *Coordination Languages and Models, Second International Conference, COORDINATION ’97, Berlin, Germany, September 1-3, 1997, Proceedings*. Ed. by David Garlan and Daniel Le Métayer. Vol. 1282. Lecture Notes in Computer Science. Springer, pp. 115–130 (page 55).

Oh, Jean, Felipe Meneguzzi, Katia P. Sycara, and Timothy J. Norman (2011). “Prognostic normative reasoning in coalition planning”. In: *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*. Ed. by Liz Sonenberg, Peter Stone, Kagan Tumer, and Pinar Yolum. IFAAMAS, pp. 1233–1234 (page 151).

Oren, N., M. Luck, S. Miles, and T. J. Norman (2008). “An Argumentation Inspired Heuristic for Resolving Normative Conflict”. In: *Proceedings of The Fifth Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems (COIN@AAMAS-08)*, pp. 41–56 (pages 28, 61, 69).

Oren, Nir (2013). “Argument Schemes for Normative Practical Reasoning”. In: *TAFA*. Ed. by Elizabeth Black, Sanjay Modgil, and Nir Oren. Vol. 8306. Lecture Notes in Computer Science. Springer, pp. 63–78 (pages 20, 23, 27, 49–52, 68, 106, 107, 114, 120, 132, 143, 145–147, 149).

Oren, Nir, Timothy J. Norman, and Alun D. Preece (2007). “Subjective logic and arguing with evidence”. In: *Artif. Intell.* 171.10-15, pp. 838–854 (page 15).

Oren, Nir, Madalina Croitoru, Simon Miles, and Michael Luck (2010). “Understanding Permissions through Graphical Norms”. In: *Declarative Agent Languages and Technologies VIII (DALT)*. Ed. by Andrea Omicini, Sebastian Sardiña, and Wamberto Weber Vasconcelos. Vol. 6619. Lecture Notes in Computer Science. Springer, pp. 167–184 (pages 27, 149).

Oren, Nir, Wamberto Vasconcelos, Felipe Meneguzzi, and Michael Luck (2011). “Acting on Norm Constrained Plans”. In: *CLIMA*. Ed. by João Leite, Paolo Torroni, Thomas Ågotnes, Guido Boella, and Leon van der Torre. Vol. 6814. Lecture Notes in Computer Science. Springer, pp. 347–363 (pages 30, 51, 151).

Preferences, Paths, Power, Goals and Norms (2013) (page 120).

Ouerdane, Wassila, Nicolas Maudet, and Alexis Tsoukiàs (2008). "Argument Schemes and Critical Questions for Decision Aiding Process". In: *Computational Models of Argument: Proceedings of COMMA 2008, Toulouse, France, May 28-30, 2008*. Ed. by Philippe Besnard, Sylvie Doutre, and Anthony Hunter. Vol. 172. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 285–296. ISBN: 978-1-58603-859-5 (page 37).

Pacheco, Natalia Criado (2012). "Using Norms to Control Open Multi-agent Systems". PhD thesis. Universidad Politecnica de Valencia (pages 13, 27–29, 60, 62, 149, 151).

Panagiotidi, Sofia, Javier Vázquez-Salceda, and Wamberto Vasconcelos (2012a). "Contextual Norm-Based Plan Evaluation via Answer Set Programming". In: *Highlights on Practical Applications of Agents and Multi-Agent Systems - 10th International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAMS 2012 Special Sessions, Salamanca, Spain, 28-30 March, 2012*. Ed. by Javier Bajo Pérez, Miguel A. Sánchez, Philippe Mathieu, Juan M. Corchado Rodríguez, Emmanuel Adam, Alfonso Ortega, María N. Moreno García, Elena Navarro, Benjamin Hirsch, Henrique Lopes Cardoso, and Vicente Julián. Vol. 156. Advances in Soft Computing. Springer, pp. 197–206 (pages 76, 99).

Panagiotidi, Sofia, Javier Vázquez-Salceda, and Frank Dignum (2012b). "Reasoning over Norm Compliance via Planning". In: *Coordination, Organizations, Institutions, and Norms in Agent Systems VIII - 14th International Workshop, COIN 2012, Held Co-located with AAMAS 2012, Valencia, Spain, June 5, 2012, Revised Selected Papers*. Ed. by Huib Aldewereld and Jaime Simão Sichman. Vol. 7756. Lecture Notes in Computer Science. Springer, pp. 35–52 (pages 31, 51, 99).

Parsons, Simon, Katie Atkinson, Zimi Li, Peter McBurney, Elizabeth Sklar, Munindar P. Singh, Karen Zita Haigh, Karl N. Levitt, and Jeff Rowe (2014a). "Argument schemes for reasoning about trust". In: *Argument & Computation* 5.2-3, pp. 160–190 (page 52).

Parsons, Simon, Nir Oren, Chris Reed, and Federico Cerutti, eds. (2014b). *Computational Models of Argument - Proceedings of COMMA 2014, Atholl Palace Hotel, Scottish Highlands, UK, September 9-12, 2014*. Vol. 266. Frontiers in Artificial Intelligence and Applications. IOS Press.

Paul A. Buckingham-Shum, Simon J. Car Kirschner (2003). *Visualizing Argumentation : Software Tools for Collaborative and Educational Sense-Making*. Ed. by Chad Carr. Computer Supported Cooperative Work. Springer (page 152).

Pednault, Edwin P. D. (1994). “ADL and the State-Transition Model of Action.” In: *Jourmal of Logic and Computation* 4.5, pp. 467–512 (page 54).

Pitt, J., D. Busquets, and R. Riveret (2013). “Formal Models of Social Processes: The Pursuit of Computational Justice in Self-Organising Multi-Agent Systems”. In: *Self-Adaptive and Self-Organizing Systems (SASO), 2013 IEEE 7th International Conference on*, pp. 269–270 (pages 13, 28).

Pokahr, Alexander, Lars Braubach, and Winfried Lamersdorf (2005). “A Goal Deliberation Strategy for BDI Agent Systems”. In: *Multiagent System Technologies, Third German Conference, MATES 2005, Koblenz, Germany, September 11-13, 2005, Proceedings*. Ed. by Torsten Eymann, Franziska Klügl, Winfried Lamersdorf, Matthias Klusch, and Michael N. Huhns. Vol. 3550. Lecture Notes in Computer Science. Springer, pp. 82–93 (pages 66, 67).

Pollock, John L. (1987). “Defeasible Reasoning”. In: *Cognitive Science* 11.4, pp. 481–518 (page 103).

— (1992). “How to Reason Defeasibly”. In: *Artificial Intelligence* 57.1, pp. 1–42 (pages 32, 33).

— (1995). *Cognitive Carpentry: A Blueprint for How to Build a Person*. Cambridge, MA, USA: MIT Press (pages 16, 21–23, 44).

Prakken, H. and G. Vreeswijk (2002). “Logics for defeasible argumentation”. In: *Handbook of Philosophical Logic, second edition, vol. 4*. Ed. by D. Gabbay and F. Guenthner. Dordrecht etc., pp. 219–318 (page 116).

Prakken, Henry (2006a). “Combining sceptical epistemic reasoning with credulous practical reasoning”. In: *Computational Models of Argument: Proceedings of COMMA 2006, September 11-12, 2006, Liverpool, UK*. Ed. by Paul E. Dunne and Trevor J. M. Bench-Capon. Vol. 144. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 311–322 (pages 25, 46, 112–114, 134).

— (June 2006b). “Formal Systems for Persuasion Dialogue”. In: *Knowl. Eng. Rev.* 21.2, pp. 163–188 (page 43).

— (2010). “An abstract framework for argumentation with structured arguments”. In: *Argument & Computation* 1.2, pp. 93–124 (pages 36, 39).

— (2012). “Some Reflections on Two Current Trends in Formal Argumentation”. In: *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*. Ed. by Alexander Artikis, Robert Craven, Nihan Kesim Cicekli, Babak Sadighi, and Kostas Stathis. Vol. 7360. Lecture Notes in Computer Science. Springer, pp. 249–272 (pages 102, 103).

Prakken, Henry and Giovanni Sartor (1997). “Argument-Based Extended Logic Programming with Defeasible Priorities”. In: *Journal of Applied Non-Classical Logics* 7.1, pp. 25–75 (pages 100, 118, 134).

Raedt, Luc De, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, eds. (2012). *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*. Vol. 242. Frontiers in Artificial Intelligence and Applications. IOS Press.

Rahwan, Iyad and Leila Amgoud (2006). “An Argumentation-Based Approach for Practical Reasoning”. In: *ArgMAS*. Ed. by Nicolas Maudet, Simon Parsons, and Iyad Rahwan. Vol. 4766. Lecture Notes in Computer Science. Springer, pp. 74–90 (pages 15, 24, 47, 48, 51, 52, 66, 118, 132, 143, 144, 146, 147).

Rahwan, Iyad and Pavlos Moraitis, eds. (2009). *Argumentation in Multi-Agent Systems, Fifth International Workshop, ArgMAS 2008, Estoril, Portugal, May 12, 2008. Revised Selected and Invited Papers*. Vol. 5384. Lecture Notes in Computer Science. Springer.

Rahwan, Iyad, Sarvapali D. Ramchurn, Nicholas R. Jennings, Peter Mcburney, Simon Parsons, and Liz Sonenberg (Dec. 2003). “Argumentation-based negotiation”. In: *Knowledge Engineering Review* 18.4, pp. 343–375 (page 44).

Rao, Anand S. and Michael P. Georgeff (1995). “BDI Agents: From Theory to Practice”. In: *In Proceedings of The First International Conference On Multi-Agent Systems (ICMAS-95)*, pp. 312–319 (pages 29, 47).

Reiter, Raymond (1991). “Artificial Intelligence and Mathematical Theory of Computation”. In: ed. by Vladimir Lifschitz. San Diego, CA, USA: Academic Press Professional, Inc.

Chap. The Frame Problem in Situation the Calculus: A Simple Solution (Sometimes) and a Completeness Result for Goal Regression, pp. 359–380. ISBN: 0-12-450010-2 (page 49).

Reiter, Raymond (2001). "On knowledge-based programming with sensing in the situation calculus". In: *ACM Trans. Comput. Log.* 2.4, pp. 433–457 (page 77).

Reiter, Raymond and Johan de Kleer (1987). "Foundations of Assumption-based Truth Maintenance Systems: Preliminary Report". In: *Proceedings of the 6th National Conference on Artificial Intelligence. Seattle, WA, July 1987*. Ed. by Kenneth D. Forbus and Howard E. Shrobe. Morgan Kaufmann, pp. 183–189 (page 76).

Riemsdijk, M. Birna van, Wiebe van der Hoek, and John-Jules Ch. Meyer (2002). "Agent Programming in Dribble: From Beliefs to Goals with Plans". In: *Formal Approaches to Agent-Based Systems, Second International Workshop, FAABS 2002, Greenbelt, MD, USA, October 29-31, 2002, Revised Papers*. Ed. by Michael G. Hinchey, James L. Rash, Walt Truszkowski, Christopher Rouff, and Diana F. Gordon-Spears. Vol. 2699. Lecture Notes in Computer Science. Springer, pp. 294–295 (pages 60, 66, 67).

Riemsdijk, M. Birna van, Mehdi Dastani, and Michael Winikoff (2008). "Goals in agent systems: a unifying framework". In: *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12-16, 2008, Volume 2*. Ed. by Lin Padgham, David C. Parkes, Jörg P. Müller, and Simon Parsons. IFAAMAS, pp. 713–720 (pages 59, 60).

Riemsdijk, M. Birna van, Mehdi Dastani, and John-Jules Ch. Meyer (2009). "Goals in conflict: semantic foundations of goals in agent programming". In: *Autonomous Agents and Multi-Agent Systems* 18.3, pp. 471–500 (page 66).

Riley, Luke, Katie Atkinson, Terry R. Payne, and Elizabeth Black (2011). "An Implemented Dialogue System for Inquiry and Persuasion". In: *Theorie and Applications of Formal Argumentation - First International Workshop, TAFA 2011. Barcelona, Spain, July 16-17, 2011, Revised Selected Papers*. Ed. by Sanjay Modgil, Nir Oren, and Francesca Toni. Vol. 7132. Lecture Notes in Computer Science. Springer, pp. 67–84 (page 44).

Sadri, Fariba, Kostas Stathis, and Francesca Toni (2006). "Normative KGP agents". In: *Computational and Mathematical Organization Theory* 2006, pp. 101–126 (pages 13, 15, 30, 47, 51).

Savage, L. (1954). *The Foundations of Statistics*. New York: Wiley (page 22).

Scheuer, Oliver, Frank Loll, Niels Pinkwart, and Bruce M. McLaren (2010). “Computer-supported argumentation: A review of the state of the art”. In: *International Journal of Computer-Supported Collaborative Learning* 5.1, pp. 43–102 (page 152).

Schulz, Claudia and Francesca Toni (2014). “Justifying Answer Sets using Argumentation”. In: *CoRR* abs/1411.5635 (pages 45, 144, 145).

Searle, John R. (2001). *Rationality in Actions*. MIT Press, Cambridge, MA, USA (pages 21, 24, 114).

Shams, Zohreh (2015). “Normative Practical Reasoning: An Argumentation-Based Approach”. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, pp. 4397–4398 (page 19).

Shams, Zohreh, Marina De Vos, and Ken Satoh (2013). “ArgPROLEG: A Normative Framework for the JUF Theory”. In: *New Frontiers in Artificial Intelligence - JSAI-isAI 2013 Workshops, LENLS, JURISIN, MiMI, AAA, and DDS, Kanagawa, Japan, October 27-28, 2013, Revised Selected Papers*. Ed. by Yukiko Nakano, Ken Satoh, and Daisuke Bekki. Vol. 8417. Lecture Notes in Computer Science. Springer, pp. 183–198 (page 19).

Shanahan, Murray and Mark Witkowski (2004). “Event Calculus Planning Through Satisfiability”. In: *Journal of Logic and Computation (JLC)* 14.5, pp. 731–745 (page 77).

Shoham, Yoav and Moshe Tennenholtz (1992). “Emergent Conventions in Multi-Agent Systems: Initial Experimental Results and Observations (Preliminary Report)”. In: *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR’92). Cambridge, MA, October 25-29, 1992*. Ed. by Bernhard Nebel, Charles Rich, and William R. Swartout. Morgan Kaufmann, pp. 225–231 (pages 13, 26).

— (1997). “On the Emergence of Social Conventions: Modeling, Analysis, and Simulations”. In: *Artif. Intell.* 94.1-2, pp. 139–166 (page 26).

Sierra, Carles, Cristiano Castelfranchi, Keith S. Decker, and Jaime Simão Sichman, eds. (2009). *8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, 2009, Volume 2*. IFAAMAS.

Simari, Guillermo Ricardo and Ronald Prescott Loui (1992). “A Mathematical Treatment of Defeasible Reasoning and its Implementation”. In: *Artif. Intell.* 53.2-3, pp. 125–157 (pages 32, 33, 102).

Sonenberg, Liz, Peter Stone, Kagan Tumer, and Pinar Yolum, eds. (2011). *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*. IFAAMAS.

Southwick, Richard W (1991). "Explaining reasoning: an overview of explanation in knowledge-based systems". In: *The knowledge engineering review* 6.01, pp. 1–19 (page 142).

Swartout, William R and Johanna D Moore (1993). "Explanation in second generation expert systems". In: *Second generation expert systems*. Springer, pp. 543–585 (page 142).

Tang, Calvin Kai Fan and Eugenia Ternovska (2005). "Model Checking Abstract State Machines with Answer Set Programming". In: *Proceedings of the 12th International Workshop on Abstract State Machines, ASM 2005, March 8-11, 2005, Paris, France*, pp. 397–416 (page 76).

— (2007). "Model Checking Abstract State Machines with Answer Set Programming". In: *Fundam. Inform.* 77.1-2, pp. 105–141 (page 76).

Tang, Yuqing and Simon Parsons (2005). "Argumentation-Based Multi-agent Dialogues for Deliberation". In: *Argumentation in Multi-Agent Systems, Second International Workshop, ArgMAS 2005, Utrecht, The Netherlands, July 26, 2005, Revised Selected and Invited Papers*. Ed. by Simon Parsons, Nicolas Maudet, Pavlos Moraitis, and Iyad Rahwan. Vol. 4049. Lecture Notes in Computer Science. Springer, pp. 229–244 (pages 44, 144).

Teach, Randy L and Edward H Shortliffe (1981). "An analysis of physician attitudes regarding computer-based clinical consultation systems". In: *Use and impact of computers in clinical medicine*. Springer, pp. 68–85 (page 142).

Thagard, Paul (2002). *Coherence in thought and action*. Cambridge: The MIT Press (page 69).

Thangarajah, John, Michael Winikoff, Lin Padgham, and Klaus Fischer (2002). "Avoiding Resource Conflicts in Intelligent Agents". In: *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI'2002, Lyon, France, July 2002*. Ed. by Frank van Harmelen. IOS Press, pp. 18–22 (page 66).

Thangarajah, John, Lin Padgham, and Michael Winikoff (2003). "Detecting & Avoiding Interference Between Goals in Intelligent Agents". In: *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, pp. 721–726 (pages 66, 67).

Thomason, Richmond H. (2000). "Desires and Defaults: A Framework for Planning with Inferred Goals". In: *Principles of Knowledge Representation and Reasoning (KR)*. Morgan Kaufmann, pp. 702–713 (page 114).

Toniolo, Alice (2013). "Models of Argument for Deliberative Dialogue in Complex Domains". PhD thesis. University of Aberdeen (pages 37, 44, 66, 67, 106, 144, 147, 151).

Toniolo, Alice, Timothy J. Norman, and Katia P. Sycara (2011). "Argumentation Schemes for Collaborative Planning". In: *PRIMA*. Ed. by David Kinny, Jane Yung jen Hsu, Guido Governatori, and Aditya K. Ghose. Vol. 7047. Lecture Notes in Computer Science. Springer, pp. 323–335. ISBN: 978-3-642-25043-9 (pages 68, 143).

— (2012). "An Empirical Study of Argumentation Schemes for Deliberative Dialogue". In: *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*. Ed. by Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas. Vol. 242. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 756–761 (pages 37, 49, 51, 52, 106, 151).

Toth, Eva Erdosne, Daniel D. Suthers, and Alan M. Lesgold (2002). "Mapping to know: The effects of representational guidance and reflective assessment on scientific inquiry". In: *Science Education* 86.2, pp. 264–286 (page 152).

Toulmin, Stephen E. (1958). *The Uses of Argument*. Cambridge University Press (pages 37, 38).

Uszok, Andrzej, Jeffrey M. Bradshaw, James Lott, Maggie R. Breedy, Larry Bunch, Paul J. Feltovich, Matthew Johnson, and Hyuckchul Jung (2008). "New Developments in Ontology-Based Policy Management: Increasing the Practicality and Comprehensiveness of KAoS". In: *9th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2008), 2-4 June 2008, Palisades, New York, USA*. IEEE Computer Society, pp. 145–152 (pages 28, 61).

Vasconcelos, Wamberto Weber, Martin J. Kollingbaum, and Timothy J. Norman (2009). "Normative conflict resolution in multi-agent systems". In: *Autonomous Agents and Multi-Agent Systems (AAMAS)* 19.2, pp. 124–152 (pages 69, 151).

Vázquez-Salceda, Javier (2003). “The role of Norms and Electronic Institutions in Multi-Agent Systems applied to complex domains. The HARMONIA framework”. PhD thesis. Universitat Politcnica de Catalunya (page 150).

Vázquez-Salceda, Javier, Huib Aldewereld, and Frank Dignum (2004). “Implementing Norms in Multiagent Systems”. In: *Multiagent System Technologies, Second German Conference, MATES 2004, Erfurt, Germany, September 29-30, 2004, Proceedings*. Ed. by Gabriela Lindemann, Jörg Denzinger, Ingo J. Timm, and Rainer Unland. Vol. 3187. Lecture Notes in Computer Science. Springer, pp. 313–327 (pages 28, 61).

Vreeswijk, Gerard (1992). “Reasoning with Defeasible Arguments: Examples and Applications”. In: *Logics in AI, European Workshop, JELIA ’92, Berlin, Germany, September 7-10, 1992, Proceedings*. Ed. by David Pearce and Gerd Wagner. Vol. 633. Lecture Notes in Computer Science. Springer, pp. 189–211 (pages 32, 33).

Vreeswijk, Gerard and Henry Prakken (2000). “Credulous and Sceptical Argument Games for Preferred Semantics”. In: *Logics in Artificial Intelligence, European Workshop, JELIA 2000 Malaga, Spain, September 29 - October 2, 2000, Proceedings*. Ed. by Manuel Ojeda-Aciego, Inman P. de Guzmán, Gerhard Brewka, and Luís Moniz Pereira. Vol. 1919. Lecture Notes in Computer Science. Springer, pp. 239–253 (pages 44, 46, 134, 135, 141).

Walker, Adam and Michael Wooldridge (1995). “Understanding the Emergence of Conventions in Multi-Agent Systems”. In: *Proceedings of the First International Conference on Multiagent Systems, June 12-14, 1995, San Francisco, California, USA*. Ed. by Victor R. Lesser and Les Gasser. The MIT Press, pp. 384–389 (pages 13, 26).

Walton, D. and E. Krabbe (1995). *Commitment in Dialogue: Basic concept of interpersonal reasoning*. Albany NY: State University of New York Press (page 43).

Walton, Douglas N. (1996). *Argumentation Schemes for Presumptive Reasoning*. L. Erlbaum Associates (pages 16, 20, 21, 36, 37, 39, 40, 106).

Walton, Douglas N., Chris Reed, and Fabrizio Macagno (2008). *Argumentation Schemes*. Cambridge University Press (pages 39, 108).

Wick, Michael R (1992). “Expert system explanation in retrospect: A case study in the evolution of expert system explanation”. In: *Journal of Systems and Software* 19.2, pp. 159–169 (page 142).

Wooldridge, Michael J. (2009). *An Introduction to MultiAgent Systems (Second Edition)*. Wiley, pp. I–XXII, 1–461. ISBN: 978-0-470-51946-2 (page 12).

Wooldridge, Michael (2000). *Reasoning about rational agents*. MIT Press, Cambridge (pages 20, 24).

Wooldridge, Michael and Wiebe van der Hoek (July 4, 2006). "On obligations and normative ability: Towards a logical analysis of the social contract". In: *Journal of Applied Logic* 3-4, pp. 396–420 (page 146).

Wooldridge, Michael and Nicholas R. Jennings (1995). "Intelligent Agents: Theory and Practice". In: *Knowledge Engineering Review* 10, pp. 115–152 (page 21).

Wooley, Bruce A. (Sept. 1998). "Explanation Component of Software System". In: *Crossroads* 5.1, pp. 24–28 (pages 14, 45).

Ye, L Richard (1995). "The value of explanation in expert systems for auditing: An experimental investigation". In: *Expert Systems with Applications* 9.4, pp. 543–556 (pages 142, 153).

Zhong, Qiaoting, Xiuyi Fan, Francesca Toni, and Xudong Luo (2014). "Explaining Best Decisions via Argumentation". In: *Proceedings of the European Conference on Social Intelligence (ECSI-2014), Barcelona, Spain, November 3-5, 2014*. Ed. by Andreas Herzig and Emiliano Lorini. Vol. 1283. CEUR Workshop Proceedings. CEUR-WS.org, pp. 224–237 (pages 20, 46, 132, 133, 143).