Rules, reasons, arguments : formal studies of argumentation and defeat

Rules, Reasons, Arguments

Formal studies of argumentation and defeat

Rules, Reasons, Arguments

Formal studies of argumentation and defeat

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit Maastricht,
op gezag van de Rector Magnificus, Prof.mr. M.J. Cohen,
volgens het besluit van het College van Decanen,
in het openbaar te verdedigen
op donderdag 5 december 1996 om 14.00 uur

door

Harry Bart Verheij

Promotor:
Prof.dr. H.J. van den Herik
Copromotor:
Dr. J.C. Hage

Beoordelingscommissie:
Prof.dr.ir. A. Hasman (voorzitter)
Prof.dr.ir. J.L.G. Dietz (Technische Universiteit Delft)
Dr. F.M.P. van Dun
Prof.dr. J.L. Pollock (University of Arizona, Tucson)
Dr. H. Prakken (Vrije Universiteit Amsterdam)

Omslag: Grietje Verheij en Charlot Luiting

ISBN 90-9010071-7

Contents

Preface

In 1991, I decided not to become an algebraic geometry researcher in Amsterdam. In 1992, I decided to become an artificial intelligence researcher in Maastricht. Both were decisions I do not regret.

Of course, things have developed very differently from what I expected then. I started with research on multimedia information retrieval, and ended with research on argumentation and defeat. Even though my research seemingly drifted away from the central goals of the ARCHIMEDES project, I have profited much from being a member of the project. It provided a broad learning environment and prevented me from involuntarily doing my research in isolation, as is so common for beginning researchers. At the frequent project meetings, I could test my research skills and get feedback on my ideas.

First of all I want to thank Jaap Hage. His constant enthusiasm and attention made learning from him and working with him a pleasure to me. He taught me many things, more than I am aware of now. By our shared attachment to the dialectical method, I could experience its fruitfulness in practice: when our opinions seemed to differ strongly, lively discussions led to better insights in the end. I owe him the name for the argumentation stages model described in the chapters 5 and 6. In the early days of that research, when he was skeptical and my ideas were not well developed, he teasingly called the stages 'wolkjes', the Dutch word for 'small clouds'. 'Clouds' led to 'cumuli', and 'cumuli' to 'CumulA'.

Second I thank Jaap van den Herik. He gave me the opportunity to become a researcher. His consistent emphasis on uncompromising self-criticism and high standards has been a lesson that I will have to teach and reteach myself in my future scientific life. He also taught me that the difficulty and importance of a good presentation of research cannot be overestimated.

I thank Gerard Vreeswijk for sharing his experiences as a researcher with me. Fate was on my side, when just after I gained interest in his work, he was appointed as a post-doc at the Department of Computer Science in Maastricht. His work paved the way for my CumulA model.

I thank Arno Lodder and Floris Wiesman for blurring the border between being colleagues and being friends. Arno's sense of humor made life and work in room 0.062 at De Nieuwenhof much better. Meeting Floris again in Maastricht after more than twenty years was a pleasant coincidence. The similarity of our backgrounds and interests still strikes me. I am also grateful to Floris for compiling the index.

I thank the other members of the ARCHIMEDES project for their attention, even at the times that my presentations were unclear and unfinished. I thank Jan Dietz and Arie Hasman for their guidance and support, and Maarten van der Meulen, Ruud van der Pol and Georges Span for companionship, especially at the harder moments. I am Georges also grateful for his kind help and flexibility during the preparation of the camera ready version of the thesis.

I thank the administrative and management staff of the Faculty of Law, Department of Metajuridica, and of the Faculty of General Sciences, Department of Computer Science. I especially want to mention Annemie Jeukens, Math Lieben and Appie Luermans.

I thank Ron Loui for inviting me to work at the Computer Science Department of the Washington University in Saint Louis (Missouri) in the summer of 1996. Over the good coffee at Euclid, we had inspiring conversations and confirmed the agreement of our views on argumentation. He gave me the freedom to finish my thesis. I thank Fernando Tohmé for his warm friendship during the hot summer in Saint Louis, and for his support in the final writing stages. I thank Nina Kang and David Packer for proofreading the thesis.

I thank the attendants of the Logic and Law working group in Utrecht for stimulating discussions, and especially Tina Smith for organizing the quarterly meetings. I thank the members and observers of the Dutch Foundation for Legal Knowledge Systems (JURIX) for the fresh and open atmosphere at the meetings and conferences.

I thank Harry van Gelderen because he showed me the strange connections between mathematics and real life by giving me a Rubik's cube. I am also grateful for his support of my 'promotie'.

I thank Aimée Herczog and Bas Keultjes for friendship, even at a distance, the Maastricht acrobats, especially my Steiger and Beez Wax friends, for fun and distraction, and Truus Hoge Verheij, Ben Verheij, Grietje Verheij, Chris Ingelse and Margreet Walinga for being there.

Acknowledgments

The research reported here is partly financed by the Foundation for Knowledge-Based Systems (SKBS) as part of the B3.A project. SKBS is a foundation with the goal to improve the level of expertise in the Netherlands in the field of knowledge-based systems and to promote the transfer of knowledge in this field between universities and business companies.

My stay with Ron Loui at the Computer Science Department at the Washington University in Saint Louis (Missouri) in the summer of 1996 was sponsored by the National Science Foundation (NSF) under grant number 9503476.

The Netherlands Organization for Scientific Research (NWO) and the Foundation for Scientific Education Limburg (SWOL) have financially supported my trips to conferences and workshops.

Guidelines for the reader

A cross-reference from one section to another section (e.g., see section 3.1) refers to a section in the same chapter, unless explicitly indicated otherwise (e.g., see chapter 2, section 3.1).

The following figure shows the interdependency of the chapters.

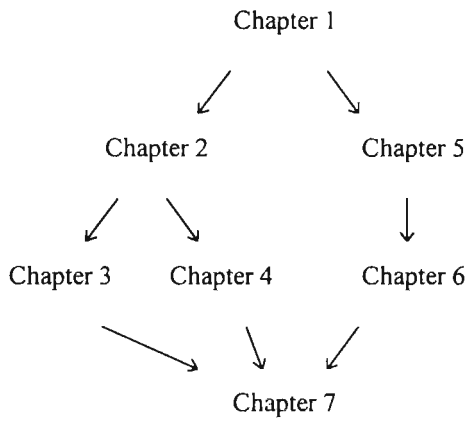

Figure 1: Interdependency of the chapters

Contact information

Bart Verheij
Department of Metajuridica
Universiteit Maastricht
P.O. Box 616
6200 MD Maastricht
The Netherlands

bart.verheij@metajur.unimaas.nl
http://www.cs.unimaas.nl/~verheij/

Chapter 1

Introduction

The subject of this thesis is argumentation. We consider argumentation as a process in which arguments supporting a conclusion are taken into account. During the process of argumentation, a conclusion originally justified by some argument can become unjustified. This is the result of the *defeasibility* of arguments.[1] Our central theme is how argumentation and the defeasibility of arguments can be formally modeled.

In this chapter, we first introduce the central concepts used throughout the thesis: argumentation (section 1), arguments (sections 2 and 3) and defeasibility (section 4). Then the research of the thesis is introduced. We put our research in perspective by giving a brief survey of recent related research (section 5), and by explaining our general aims and biases (section 6). The introductory chapter concludes with the research goals and method (section 7), and an outline of the thesis (section 8).

1 The process of argumentation

Argumentation is a process.[2] Its purpose is to justify conclusions (see, e.g., Pollock, 1987). Which conclusions are justified changes during the argumentation process. For instance, let us consider a story about John. It starts as follows.

> John is going to work and notices that it is still freezing. He sees some people skating on the lake that he passes each day, and he realizes that the ice is finally thick enough. After his arrival at the office, he notices that his colleague Mary is not there, and wonders whether she has taken a day off. Later that morning, he meets Harry at the coffee machine. Harry tells John that whenever the ice is

---

1 The term 'defeasibility' was introduced by Hart in 1948 (cf. Loui, 1995a).

2 This is an old idea in philosophy and can already be found in the work of Aristotle (cf. Rescher, 1977). Recently, the importance of process for argumentation has been reemphasized, e.g., by Loui (1992), Vreeswijk (1993) and Lodder (1996).

An argument from premises to conclusion consists of one or more steps. For instance, the following argument that John cannot finish his work consists of two steps.

The ice is thick enough for skating.
Whenever the ice is thick enough for skating, Mary takes a day off to go skating.
So, Mary takes a day off to go skating.
Whenever Mary takes a day off, John cannot finish his work.
So, John cannot finish his work.

This argument has three premises. Two of the premises are used in the first step of the argument, and support the intermediate conclusion that Mary takes a day off. The third premise is used in the second step of the argument to support the conclusion that John cannot finish his work. The structure of the argument is shown in Figure 2. It shows the premises, the intermediate conclusion, and the conclusion of the argument.

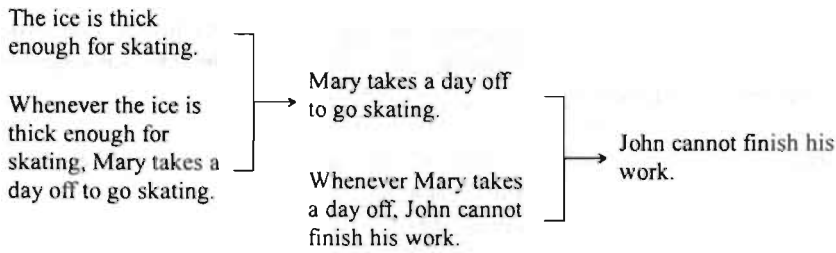

Figure 2: A two step argument

In the story, it turns out that John's conclusions that he cannot finish his work and that Mary takes a day off are in the end not justified. The arguments are defeated because of Anne's prohibition. The defeasibility of arguments is our central theme.

Summarizing, we treat arguments as reconstructions of how conclusions are supported. We regard argumentation as the process of collecting arguments in order to justify conclusions. A property of the process of argumentation is that whether arguments justify their conclusions can change during the process. They can become defeated. Arguments that at an early stage in the argumentation process justify their conclusion do not necessarily justify it at a later stage.

3 Arguments and proofs

In this thesis, we deal with formal models of argumentation. A well-known formal model of argumentation is the proof theory of classical deductive logic (in its various guises: Propositional Logic, First-Order Predicate Logic, Modal Logic).[3] Proof theory deals with proofs, that in some ways resemble arguments.

For instance, a proof that resembles the argument in Figure 1 is

$$\frac{\text{Thick-ice} \qquad \text{Thick-ice} \rightarrow \text{Day-off}}{\text{Day-off}}$$

Informally, a proof is a series of proof steps starting from given premises. Proof steps are instances of deduction rules. The example consists of one proof step that is an instance of the deduction rule known as Modus Ponens:

$$\frac{\textit{Sentence-1} \qquad \textit{Sentence-1} \rightarrow \textit{Sentence-2}}{\textit{Sentence-2}}$$

Here *Sentence-1* and *Sentence-2* are any two sentences of the logical language.

The similarity of proofs and arguments is clear. For instance, the structure of proofs is closely related to the structure of arguments. Like an argument, a proof supports its conclusion. Like an argument, a proof can consist of several steps from premises to conclusion.

In one respect, however, proofs differ from arguments: arguments are defeasible. Additional information may have the effect that an argument does no longer justify its conclusion and becomes defeated. This does not hold for proofs in deductive logic. Additional proofs never make other proofs unacceptable. Therefore, the proof theory of deductive logic is inappropriate as a model of argumentation. In this thesis, formal models of argumentation are discussed that can deal with the defeasibility of arguments.

4 The defeasibility of arguments

In this section, we informally discuss four cases in which arguments may become defeated. These are meant as illustrations of the defeasibility of arguments, and not as a taxonomy of types of defeasibility.

4.1 Exceptions to rules

So far, we have seen examples of arguments, but we have not yet investigated how the steps in an argument arise. We reconsider our example.

---

3 Lukaszewicz (1990) and Gabbay *et al.* (1993) give overviews.

The ice is thick enough for skating. Whenever the ice is thick enough for skating, Mary takes a day off to go skating.
So, Mary takes a day off to go skating.

This is an argument that consists of a single step. The argument above is an instance of the following scheme:

Situation-1. Whenever *Situation-1*, *Situation-2*.
So, *Situation-2*.

Not only the argument above, but all instances of this scheme are arguments. There is some kind of relation between the premises and the conclusion of the argument. This relation is called a *rule*.[4] If a rule gives rise to acceptable arguments, it is valid.

The rule behind the argument scheme above is closely related to the deduction rule Modus Ponens of classical deductive logic (see section 3). If an instance of a (valid) rule is used as a step in an argument, we say that the rule is applied.

A characteristic of rules is that they can have exceptions: the conclusion of a rule does not always follow if its condition is satisfied. In the case of an exception to a rule, arguments that contain a step warranted by that rule are defeated.

We have already seen an exception to the rule above, namely the case that Mary's boss prohibited her to take a day off. In such a case the rule is not applied. Exceptions to the rule can exist, since even if Mary normally goes skating when the ice is thick enough, there can be other reasons why she does not go.

4.2 Conflicting arguments

If arguments have incompatible conclusions, we speak of conflicting arguments. For instance, Mary can have a reason to go to work, and at the same time a reason to take a day off. Not going to work may cause problems at the office, but not taking the day off means that she misses one of the few opportunities to go skating. So, Mary might consider the following two arguments :

There will be problems at the office, if I take a day off.
So, I go to work.

---

[4] One might think that 'Whenever the ice is thick enough for skating, Mary takes a day off to go skating' is a rule. In the argument in the text it is however a premise of the argument. If it is considered to be a valid rule, it gives rise to the following argument:

The ice is thick enough for skating.
So, Mary takes a day off to go skating.

We come back to this difference in chapter 4, section 1, where we discuss syllogistic and enthymematic arguments.

I miss one of the few opportunities to go skating, if I go to work.
So, I take a day off.

However, it is impossible to go to work and to take a day off, so the conclusions of Mary's two arguments are incompatible, and the arguments are conflicting.

In the arguments above, 'There will be problems at the office, if I take a day off' is a reason for 'I go to work', and 'I miss one of the few opportunities to go skating, if I go to work' is a reason for 'I take a day off'. In general, we call the direct predecessor of a conclusion in an argument a *reason* for that conclusion. In a case such as in this example, where reasons support incompatible conclusions, we say that the reasons are conflicting.

Conflicting arguments must be distinguished from contradictory proofs in deductive logic. If two proofs are contradictory, anything can be proven, and there must be a false premise. If two arguments conflict, there is not necessarily a false premise. It can also be the case that one (or both) of the arguments should be considered defeated. In the example above, probably both arguments are defeated, and replaced by an argument in which Mary takes her preferences into account:

There will be problems at the office, if I take a day off.
I miss one of the few opportunities to go skating, if I go to work.
Opportunities to go skating are extremely rare, and the problems can be solved tomorrow.
So, I take a day off.

4.3 Conclusive force

Not all arguments support their conclusion equally well; arguments have different degrees of conclusive force. Some arguments make their conclusion more plausible than others ('If it was Mary who told you John is nice, I believe he is. If Anne told you, I don't know'). If an argument uses statistical evidence, one conclusion can be more probable than another ('John's boss is probably male') If the conclusive force of an argument is too weak, it is defeated.

The depth of an argument influences its conclusive force: a series of argument steps is often less cogent than one step. For instance, the argument that there will be problems at the office is less cogent than the shorter argument that Mary takes a day off. The conclusive force becomes less because the larger argument can be defeated by exceptions to *both* argument steps.

In the story there is an exception to the first argument step. The conclusion that Mary has taken a day off is not supported, because Anne prohibited it. As a result, the conclusion that there will be problems at the office is then also no longer justified. But if it was justified to believe that Mary took a day off, there could be still be an exception to the second step. For instance, if a temporary employee is hired, it is not justified to believe that there will be problems at the office. We call this the *sequential weakening* of an argument.

Also the number of arguments that support a conclusion or intermediate conclusion influences the conclusive force of an argument. An argument that contains more (independent) reasons for some conclusion can become more cogent. For instance,

Harry told you John is nice; Pat told you John is nice.
So, I believe he is.

can justify its conclusion, while

Harry told you John is nice.
So, I believe he is.

may not. We call this the *parallel strengthening* of an argument.

4.4 Other arguments taken into account

Whether an argument is defeated is influenced by the other arguments taken into account. We have already seen an example: the argument that Mary takes a day off to go skating is defeated as soon as there is another argument that justifies the conclusion that Mary's boss forbids her to. In our story the latter argument was actually a statement: it did not contain an argument step.[5]

In the example, there is an exception to the rule that Mary takes a day off to go skating if the ice is thick enough. The argument that Mary takes a day off can also be defeated by an argument that explicitly takes the exception into account:

The ice is thick enough for skating.
If the ice is thick enough for skating and Mary's boss does not forbid her to take a day off, Mary takes a day off to go skating.
If the ice is thick enough for skating and Mary's boss forbids her to take a day off, Mary does not take a day off to go skating.
Mary's boss forbids her to take a day off.
So, Mary does not take a day off to go skating.

Another example of the influence of arguments on each other is that arguments can challenge each other. We say that one argument challenges another argument if the challenged argument is defeated in case the challenging argument is not. For instance, the argument

John dislikes Mary.
So, I think that Mary is not nice.

---

5 By convention, we treat statements as arguments with trivial structure (cf. chapter 5, section 2.1).

might be challenged by the statement:

John and Mary had a relationship, and Mary finished it.

5 Related research

In the previous sections, we introduced the central concepts of the thesis. We continue with an introduction to the research in the thesis, and start with a brief survey of related research.

Recently there has been a revival of research on argumentation and defeat. This revival has been motivated by several cross-disciplinary interests. For instance, the following - not necessarily disjoint - disciplines have stimulated the research on argumentation and defeat:

- *Logic*: The research on nonmonotonic logics remains popular (Gabbay *et al.* (1994b) give an overview) and now encompasses the defeasibility of arguments as a special topic (cf. Nute, 1994).
- *Computer science*: The computational complexity of nonmonotonicity attracted the attention of the logic programming community (e.g., Dung, 1993, 1995; Bondarenko *et al.*, 1993).
- *Artificial intelligence*: Since reasoning with defeasible arguments seems to lead to successful behavior of people, artificial intelligence researchers try to capture its essence (e.g., Nute, 1988; Geffner and Pearl, 1992; Simari and Loui, 1992).
- *Epistemology*: Questions about the justification and support of beliefs have resulted in formal epistemological theories (e.g., Loui, 1987, 1991; Pollock, 1987-1995).
- *Argumentation theory*: Notions such as counterargument and reinstatement have been formally studied (e.g., Vreeswijk, 1991, 1993; Verheij, 1995a, b, c).
- *Dialectics*: Several game-like formalisms have been proposed in which two parties are disputing an issue (e.g., Loui, 1992; Gordon, 1993a, 1993b, 1995; Vreeswijk, 1993; Brewka, 1994; Leenes *et al.*, 1994; Hage *et al.*, 1994; Lodder and Herczog, 1995).
- *Legal theory*: The pragmatic solutions in legal reasoning to deal with exceptions and conflicts have inspired researchers and have been formally analyzed (e.g., Hage, 1993, 1995; Prakken, 1993a, b, 1995; Sartor, 1994).

This brief survey contains only a selection of recent research to give an idea of the current activity and the diversity of perspectives. Overviews of research on argumentation and defeat have recently been given by Bench-Capon (1995), who focuses on artificial intelligence and law, and Loui (1995b), who focuses on computational dialectics.

6 General aims and biases of research

Research on the modeling of argumentation can have rather different aims, for instance:

- to describe and evaluate actual human argumentation by means of empirical investigation, e.g., in cognitive science or psychology;
- to apply an argumentation model in order to build intelligent computers and programs, e.g., in computer science and artificial intelligence;
- to investigate and enhance our conceptualizations of argumentation in order to better understand its nature, e.g., in philosophical and mathematical logic.

Of course, doing research on the modeling of argumentation, one does not normally have only one of the aims above: even if one is mostly interested in intelligent computer programs, one can be inspired by actual human argumentation, and be led to the enhancement of one's initial model of argumentation. Nevertheless, research is often biased towards one or more of the mentioned aims of study.

In Figure 3, we have visualized the biases of some research on the modeling of argumentation in a triangular diagram. The three corners of the triangle correspond to the three aims mentioned, and are suggestively labeled 'Minds and humans', 'Machines and programs', and 'Theories and models'. Researchers or subjects of research are indicated by a labeled dot.[6] The closer a dot is to one of the corners, the more the corresponding researcher or subject of research is biased towards the aim of study of that corner.[7]

Some research is mostly biased to one of the three aims. We give examples of each. First, we mention First-Order Predicate Logic.[8] As a model of argumentation, it is most appropriate as a theoretical model, due to its nice mathematical properties, but less as an empirical model or as a computational model. It is therefore indicated in the upper corner. Second, Van Eemeren and Grootendorst (Van Eemeren *et al.*, 1981, 1987) have provided a model of argumentation explicitly meant to analyze argumentation as it occurs in argumentative texts, and are therefore indicated in the lower-left corner. Third, the research on logic programming is clearly mostly aimed at building intelligent machines, notwithstanding its theoretical achievements, and is therefore indicated in the lower-right corner.

---

6 Several of the indicated researchers or subjects of research are extensively discussed later on.

7 The triangle has *barycentric coordinates*. One can think of the triangle as the set of points (x, y, z) in the plane $x + y + z = 1$, such that $0 \leq x \leq 1$, $0 \leq y \leq 1$ and $0 \leq z \leq 1$. For instance, the corners of the triangle are the points where one of the coordinates is equal to 1. The sides of the triangle are the points where one of the coordinates is equal to 0. The values of each of the three coordinates represent the bias level towards one of the corners.

8 Van Dalen (1983) and Davis (1993) give introductions to First-Order Predicate Logic.

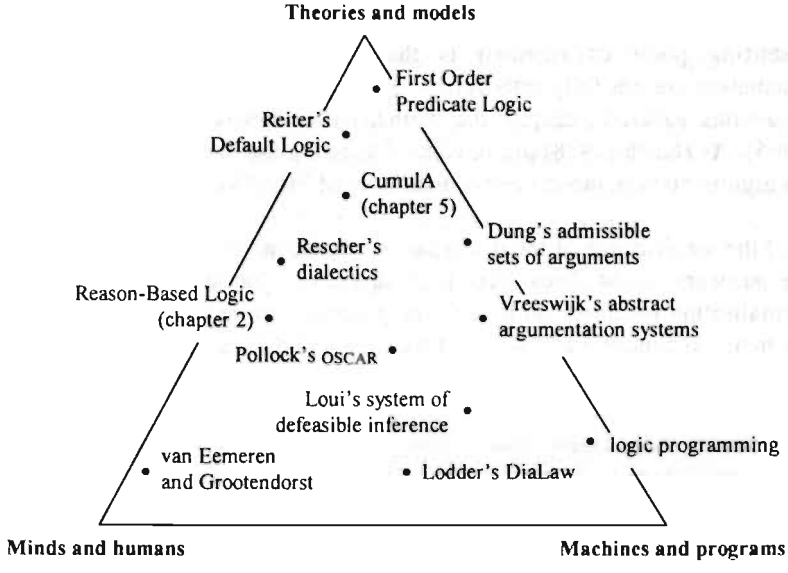

Figure 3: Biases diagram of some research on modeling argumentation

Some research is equally biased towards two of the aims, and is therefore indicated near the middle of one of the sides of the triangle. For instance, Vreeswijk's abstract argumentation systems (1991, 1993) were meant both as a model for theoretical study and for computational application.[9] Pollock's (1995) research on OSCAR is indicated in the middle of the triangle, since it equally contains elements of all three aims: Pollock has applied his philosophical theories on epistemology in the computer program OSCAR that is designed to argue as people do (or should do).

In order to show our aims of research, we have included our two main topics, Reason-Based Logic and CumulA, in the triangle. The first, Reason-Based Logic (see chapter 2), is indicated near the middle of the left-side of the triangle. It was inspired by actual human argumentation, especially in the field of law (see chapter 4), but it was also developed in order to compare it with other models. The second main topic of research, CumulA (see chapter 5), is indicated near the upper corner, since it was mainly designed as an abstract model of argumentation, that can be used to analyze different approaches towards modeling argumentation.

Although the diagram is merely tentative and we make no claim about its 'truth', we nevertheless hope that the biases diagram illustrates how differently biased research on modeling argumentation can be, and what the biases of our own research are.

---

[9] Vreeswijk (1995) describes the program IACAS, which was written to demonstrate his abstract argumentation systems.

7 Research goals and method

Our starting point of research is that the currently available models of argumentation are not fully satisfactory. This starting point, although certainly not new, remains valuable, despite the abundance of newly presented models (see section 5). As Haack (1978) put it, when discussing the paradigmatic example of a rule in argumentation, the material conditional of First-Order Predicate Logic,

> '(...) the significance of the discrepancies between 'if' and '→' will depend on the answers to at least two (...) questions: for what purpose(s) is the formalisation intended? and, does that purpose require something stronger than the material conditional? Both (...) are deep and difficult questions.'
> (Haack, 1978, p. 38)

The recent revival of research (cf. section 5) is partly due to a new answer to the first of these questions: recent research often is concerned with defeasible arguments, leading to other formalizations of rules. As for rules, new purposes of formalizing argumentation, such as capturing the role of counterarguments and of the process character of argumentation, lead to new models.

The purpose of our research is to find answers to two groups of research questions.

- What is the role of rules and reasons in argumentation with defeasible arguments? What properties of rules and reasons are relevant for argumentation and defeat? How do these properties relate?
- What is the role of process in argumentation with defeasible arguments? How is the defeat of an argument determined by its structure, counterarguments and the argumentation stage?

Trying to answer these groups of questions, we study argumentation and defeat from two angles, resulting in formalisms of different nature, Reason-Based Logic and CumulA.

Reason-Based Logic is a model of the nature of rules and reasons, which are at the basis of argumentation. We investigate the properties of rules and reasons that are relevant for the argumentation and defeat, and how these properties relate to each other. This part of the research is joint work with Hage, who initiated the development of Reason-Based Logic (see chapter 2).

CumulA is a model of argumentation in stages. We investigate how the structure of an argument is related to defeat, when arguments are defeated by counterarguments, and how the status of arguments is affected by the argumentation stage.

The thesis has five goals:

- Providing a model of rules and reasons, Reason-Based Logic, focusing on properties that are relevant for the defeasibility of arguments.
- Demonstrating the usefulness of the model by providing examples in the field of law.
- Discussing how Reason-Based Logic relates to previously proposed models.
- Providing a model of argumentation, CumulA, that focuses on the process of taking arguments into account, and shows how the status of an argument is determined by the structure of the argument, the counterarguments and the stage of the argumentation process.
- Demonstrating how CumulA can be used to analyze other models of argumentation.

Our method of research can be summarized, as follows:

> Developing formal models of argumentation on the basis of informal examples.

The advantage of formal models is that they are clear and precise, which is necessary to show the intentions of the model and is useful for revealing errors and shortcomings. A drawback of formal models, as put forward by Van Eemeren *et al.*, discussing the attraction of Toulmin's less formal model (Toulmin, 1958), is that:

> 'Studying formal logic systems requires quite a lot of effort, its relevance for practical purposes is not immediately apparent and the return on the effort spent is slight.' (Van Eemeren *et al.*, 1987, p. 206)

This is felt so by many people, and indeed the feeling seems to be justified by the research on nonmonotonic logics, which has become a mathematically inclined subject, even though it was initially inspired by intuitive examples.

The drawback can partly be circumvented by providing informal examples. We not only do this to make the text more legible, but also as an essential ingredient of our method: without informal examples, a formalism remains uninterpreted, and therefore much less useful. We are backed by Haack (1978), who in her 'Philosophy of logics' stresses the importance of informal interpretation and extra-systematic judgments (p. 32ff.) for devising and evaluating a formal model.

As a result, in this thesis, we stick to the precision and rigor of formal models, but precede all formal definitions by informal examples, needed to interpret the formalism.

8 Outline of the thesis

The structure of this thesis follows the research goals discussed in the previous section.

In chapter 2, we describe Reason-Based Logic. We determine types of facts concerning rules and reasons that are relevant for the defeasibility of arguments, and show their relations. Using this semantics of rules and reasons, we determine some intuitively attractive modes of reasoning. However, these lead to the difficulties of nonmonotonic reasoning. We show how the ideas of Reiter (1980, 1987) can be used to define rigorously which conclusions nonmonotonically follow from a given set of premises.

Chapter 3 contains a series of examples of Reason-Based Logic, taken from the field of law. We give applications of Reason-Based Logic to the theory of legal reasoning: we describe three different ways of reconstructing reasoning by analogy, and provide an integrated view on rules and principles, which seem fundamentally different (cf. Dworkin, 1978, p. 22ff. and 71ff.).

In chapter 4, we survey other models of rules, and compare them to Reason-Based Logic. We do this by treating a number of issues concerning the formalization of rules, and discussing various approaches to deal with these issues.

In chapter 5, the second part of the thesis starts with a discussion of CumulA. It is a formal model of argumentation with defeasible arguments, focusing on the process of taking arguments into account. The main ingredients of the formalism are arguments, defeaters, argumentation stages and lines of argumentation.

In chapter 6, we show how CumulA can be used to analyze models of argumentation. We investigate types of argument structure and of defeat, the role of inconsistency and counterarguments for defeat, and directions of argumentation. As a result, we are able to distinguish a number of argumentation theories (based on existing argumentation models) on formal grounds.

The thesis ends with the results and conclusions of the research (chapter 7). We also give some suggestions for future research.

Chapter 2

Reason-Based Logic: a semantics of rules and reasons

In this chapter, a formalism is developed that models rules and reasons. The formalism, called Reason-Based Logic, is a formal semantics of rules and reasons: Reason-Based Logic specifies the types of facts concerning rules and reasons that are relevant for the defeasibility of arguments, and makes the relations that must hold between these facts precise.[1] Examples of such facts are the fact that some rule applies, or that certain reasons outweigh other reasons. A crucial difference with other logical formalisms is that Reason-Based Logic provides a semantics in which such facts and their relations are made explicit.

The chapter begins with a motivation of the approach by means of examples (section 1). After a discussion of what is meant by a formal semantics (section 2), the formalism is introduced using the informal examples (section 3). Then a description of the formalism follows. First the types of facts concerning rules and reasons, as distinguished in Reason-Based Logic, are described (section 4), and second the relations between these types of facts (section 5). Third we define which conclusions follow from given premises (section 6).

1 Rules and reasons by example

In the previous chapter, we introduced argumentation by concentrating on the arguments that can justify a conclusion, and their defeasibility. In this chapter, we focus on rules and reasons. Both are fundamental for argumentation: *rules* give rise to the *reasons* that are used in arguments to support a conclusion. We start with a

---

[1] Hage initiated the development of Reason-Based Logic; it was continued in close cooperation with Verheij. Hage (1991) describes a theory of rational belief, called Reason Based Reasoning, that already contains the basic informal ideas of Reason-Based Logic. Verheij (1994) describes a limited version of Reason-Based Logic to get the formalism right. Hage and Verheij (1994) describe the first full version of Reason-Based Logic that is also formally satisfactory. The description of Reason-Based Logic in this chapter as a specification of types of facts concerning rules and reasons and the relations between these facts is related to that of Verheij (1995e). See also note 14.

number of informal examples. Issues related to the defeasibility of arguments (introduced in chapter 1, section 4) are examined in detail.

1.1 Rules and reasons

> Mary and John are planning to have a picnic on Sunday. The evening before, they watch the weather report on television. According to the weather report, it is going to rain the whole day. Mary and John are disappointed.

Mary and John's disappointment is the result of the following argument:

> According to the weather report, it will rain all day.
> So, it will rain all day.

Because of this argument John and Mary conclude that it will rain all day. Their conclusion is in this argument supported by the prediction in the weather report. 'According to the weather report, it will rain all day' is a reason for 'It will rain all day'.

John and Mary would have made a similar argument if the prediction in the weather report had been different. If the prediction had been that it would be a sunny day, John and Mary would have concluded that it will be sunny because of the weather report.

So, reasons do not arise individually, but follow a pattern. The prediction of the weather report gives rise to a reason, whatever that prediction is, in the following pattern:

> According to the weather report, it will *be weather type so-and-so*.
> So, it will *be weather type so-and-so*.

Each instance of this argument scheme can be an argument that supports the conclusion that it will be some type of weather. Moreover, each instance can be a step in a larger argument. The relation between a reason and a conclusion as expressed by such an argument scheme is what we call a *rule*.[2] If an instance of the scheme can actually be used as part of an argument that supports its conclusion (for instance, when its condition holds) we say that the rule *applies*.

Not all rules give rise to argument schemes that lead to acceptable arguments. We consider a rule to be *valid* if it is generally accepted (in some reasoning community) that the application of the rule can give rise to an argument that supports its conclusion.[3]

---

[2] Rules in Reason-Based Logic correspond to warrants in Toulmin's (1958) argument scheme.

[3] This is in contrast with the *legal validity* of a rule, which requires that the rule is obtained by a legal procedure, such as when the rule is made by the legislator, and approved by the parliament.

1.2 Exclusionary reasons

> After watching the weather report, John is disappointed. He would not like to have a picnic if it is going to rain all day. Mary smiles, and says that he does not have to worry, because the weather report on national television is not good at predicting the weather in their district, due to the peculiar local circumstances. Therefore, there is no reason to conclude that it will rain all day.

The story illustrates the defeasibility of arguments, introduced in the previous chapter. In chapter 1, the first example of the defeat of an argument was an exception to a rule (chapter 1, section 4.1): in exceptional circumstances the conclusion of a rule does not follow, even though its condition holds.

In the story about John and Mary, we again encounter an example of an exception: the weather report on national television is not good at predicting the local weather. Therefore, the fact that, according to the weather report, it will rain all day is *not* a reason that it will rain all day in this district. The rule underlying the argument scheme

> According to the weather report, it will *be weather type so-and-so*.
> So, it will *be weather type so-and-so*.

is not applicable, even though its condition is satisfied by the fact that, according to the weather report, it will rain all day. We say that 'The weather report on national television is not good at predicting the local weather' is an *exclusionary reason* to the applicability of the rule.[4] In case there is no exclusionary reason to the applicability of a rule, the rule is applicable. We will later see that even an applicable rule does not always apply, although it normally does (section 1.4).

1.3 Weighing reasons

> That Saturday evening, John's father pays a visit, and the plan to have a picnic is discussed. He agrees that the weather report on television is not good at predicting the local weather, but says that he nevertheless thinks that it will rain on Sunday. Because John's father has been a farmer for more than twenty years, John and Mary take his opinion seriously. They go to bed disappointedly. The next morning Mary looks out the window and sees that the sky is completely cloudless. She happily tells John that it might not rain after all.

---

[4] Our use of the term 'exclusionary reason' is closely related to Raz's (1990, p. 35ff.). Raz focuses on reasons for acting, and he defines an exclusionary reason as a reason not to act for some other reason. Our exclusionary reasons are reasons that make a rule inapplicable, even in case its condition is satisfied.

At this point in our story, John and Mary can make two arguments, one that it will rain:

> John's father thinks that it will rain.
> So, it will rain.

and the other that it will not rain:

> The sky is completely cloudless.
> So, it will not rain.

This is an instance of conflicting arguments (see chapter 1, section 4.2). Because there is a reason that it will rain, and also a reason that it will not rain, John and Mary can currently not draw a conclusion.

> At breakfast, John says he is at a loss, and does not know what to think about the weather. He still takes his father's opinion seriously, but agrees with Mary that the weather looks very good. After some discussion, John and Mary decide that what they see with their own eyes provides the stronger reason, and they conclude it will not rain.

In the story, John and Mary have *weighed* the conflicting reasons.[5] Since John and Mary consider the second reason the strongest, the argument

> The sky is completely cloudless.
> So, it will not rain.

justifies its conclusion, while the argument

> John's father thinks that it will rain.
> So, it will rain.

does not, and John and Mary conclude that it will not rain.

Weighing can involve several reasons for and against a conclusion. If, for instance, the prediction of the national weather report had been good at predicting the local weather, and therefore the rule based on the prediction was not excluded, there would have been an additional reason that it will rain. In that situation, the reasons would again have to be weighed. John and Mary might still decide that what they see with their own eyes gives a reason that is strong enough to outweigh both opposing reasons, but they might also change their opinion and decide that the reasons provided by the weather report and the opinion of John's father together are stronger than the cloudless sky alone.

---

5 Cf. Naess (1978), p. 100ff.

1.4 Reasons concerning the application of a rule

> After preparing the food, John and Mary drive off to their favorite picnic site. They turn their local radio station on, and at ten o'clock the weather report brings bad news: after a nice start of the day, it will begin raining before noon. John and Mary know that, in contrast with the national weather report on television, this local weather report provides a strong reason that it will rain. Nevertheless, they refuse to take it into account, against better judgment.

John and Mary's seemingly irrational behavior has a reason: otherwise, they would have to conclude that it will rain, and they would certainly not enjoy their trip any longer. As before, John and Mary consider the rule underlying the argument scheme below to be valid:

> According to the weather report, it will *be weather type so-and-so*.
> So, it will *be weather type so-and-so*.

In the case of the report on television, this rule was excluded, because the national report is not good at predicting the local weather. This exclusionary reason does not hold for the local weather report on the radio. Nevertheless, John and Mary do not take the reason that it will rain into account. In other words, they do not apply the rule.

Nevertheless, they have a reason for applying the rule since the rule is applicable: the condition of the rule is satisfied, and the rule is not excluded. They also have a reason against applying the rule since if they would apply it they would certainly not enjoy their trip any longer. Their arguments are the following:

> The rule's condition is satisfied.
> So, the rule applies.

and

> The trip will certainly not be enjoyable any longer if the rule is applied.
> So, the rule does not apply.

Again there is a conflict of reasons, and the reasons have to be weighed. In this case, John and Mary consider the reason not to apply the rule to be the strongest.

The seemingly irrational behavior of Mary and John shows an important characteristic of rule application: it is an act, and there can be reasons for and against performing the act. Their behavior is only *seemingly* irrational: John and Mary do have a reason not to apply the rule.[6]

---

[6] In chapter 3, section 5.2, another example of reasons against the application of a rule is discussed taken from the field of law.

And, for those who may wonder, Mary and John's behavior *did* have the right result: the weather stayed well during their picnic, and they had a nice afternoon. Only when they got back in their car, did it begin to rain heavily.

1.5 Overview

In the remainder of this chapter we will forget about the actual practice of argumentation (to which we will return in chapter 5), and focus on the rules and reasons on which argumentation is based. The resulting model of rules and reasons can be used to analyze argumentation. As we hope our examples have shown, such a model is bound to be rather complicated.

In the examples, we made the following points about rules and reasons:

- Reasons for a conclusion do not arise individually, but follow a pattern represented by a valid rule.
- By the application of a rule, a reason arises that supports a conclusion in an argument.
- A rule can be excluded if there is an exclusionary reason. An excluded rule is not applicable, even if its condition is satisfied.
- In case of conflicting reasons, whether a conclusion follows depends on how the reasons pro and con are weighed. The outcome of the weighing can change if new reasons arise.
- The application of a rule is an act. There can be reasons for and against performing the act. If a rule is applicable, the fact that makes it applicable is a reason to apply the rule.

The remainder of this chapter is devoted to the elaboration of these points and to the development of a formalism called Reason-Based Logic that is based on them.

2 Semantics

In the previous section, we have informally introduced our view on the role of rules and reasons in argumentation with defeasible arguments, by means of examples. This view is at the core of Reason-Based Logic. Using these examples, we develop the formalism Reason-Based Logic in the subsequent sections, in accordance with our method of research (chapter 1, section 7). Reason-Based Logic can be regarded as a formal semantics of rules and reasons. In this section, we explain what we mean by this.

We introduce some convenient terminology. In the world there are *facts*. For instance, it can be a fact that the earth is round and that there is an oak tree in the park. Facts can be expressed by *sentences* in some language. For instance, the fact that the earth is round can be expressed in English as 'The earth is round' and in Dutch as 'De aarde is rond'. Not all sentences express facts. For instance, if it is a

fact that the earth is round, then 'The earth is flat' does not express a fact. A sentence that expresses a fact is *true*. We call a part of the world that is expressed by a sentence, whether it is true or not, a *state of affairs*. Both sentences 'The earth is round' and 'The earth is flat' express states of affairs, but only one of them can express a fact.

Not all facts deal with physical objects, such as the earth or oak trees. In this chapter, for instance, we are particularly interested in objects related to argumentation, such as rules and reasons. It can be a fact that one reason outweighs another reason, or that there is an exception to a rule.

Facts are not isolated, but stand in relation to each other. An example is the combination of facts by conjunction: it is a fact that the earth is round and it is a fact that there is an oak tree in the park if and only if it is a fact that the earth is round and there is an oak tree in the park. If we look at the corresponding sentences, we obtain the following:

> The sentence 'The earth is round' is true and the sentence 'There is an oak tree in the park' is true if and only if the sentence 'The earth is round and there is an oak tree in the park' is true.

We give another example, that is related to argumentation: if it is a fact that Mary's argument justifies its conclusion, then it is also a fact that there are applying rules that give rise to the steps in Mary's argument.

We call a specification of the types of facts in some domain and the relations that hold between these facts a *semantics* of that domain.[7] Since facts can be expressed as sentences of some language, the types of facts in a domain are specified by defining an appropriate language. The relations that hold between facts are specified in terms of relations between the truth values of sentences.

A well-known example is the 'domain of the logical connectives', and its well-known Tarski semantics.[8] One of the types of facts in this domain is conjunction. In terms of sentences, '*S1* and *S2*' expresses the conjunction of the facts expressed by '*S1*' and '*S2*'. The relation that holds between facts combined by conjunction is, in terms of the corresponding sentences:

> '*S1*' is true and '*S2*' is true if and only if '*S1* and *S2*' are true.

For the other logical connectives, similar relations hold.

---

[7] We use this terminology in analogy with that of the Tarski semantics in formal logic (see e.g., Davis, 1993, p. 34ff.). However, in style the semantics of rules and reasons discussed in this chapter differs, and is comparable to the representations of the commonsense world, as discussed by, e.g., Hayes (1985), Hobbs and Moore (1985), and Davis (1990).

[8] See Haack (1978, p. 108ff.) for a philosophical account, or any introductory text on formal logic, e.g., Davis (1993, p. 34ff.), for a formal account.

In the domain of the logical connectives, the truth value of a sentence is determined by the truth values of its building blocks, such as in the example of conjunction. The logical connectives are said to be truth-functional. In other domains this is not always the case. For instance, the truth value of the sentence 'John loves Mary' depends on the truth value of the sentence 'John hates Mary'. In a semantics of love and hate this relation has to be specified. The semantics might for instance state that 'John loves Mary' and 'John hates Mary' cannot both be true.

In this chapter, we describe a semantics of the domain of rules and reasons. Also in this domain the truth value of sentences is not solely determined by the truth value of their building blocks. An example of a relation between truth values of sentences in this domain is:

> If 'The rule with conclusion *conclusion* and condition *condition* is excluded' is true, then 'The rule with conclusion *conclusion* and condition *condition* is valid' is true.

Here *condition* and *conclusion* are variables that stand for the condition and conclusion of a rule. Informally, this relation between true sentences says that only valid rules can be excluded.

The above shows a second difference between the domain of rules and reasons and the domain of the logical connectives: there exists a semantics of the logical connectives that is generally agreed upon, namely the Tarski semantics. This semantics is so well-known that it seems to be the obviously right one, and even a 'silly pedantic exercise' (Davis, 1993, p. 34). In the domain of rules and reasons, however, this is not the case. There is no general agreement on the elementary concepts nor on their relations. This adds to the importance of our method of research: any attempt to describe a semantics of rules and reasons should be accompanied by informal examples (cf. chapter 1, section 7).

The validity of rules and the existence of reasons are the bottom line of our treatment of argumentation: our semantics of rules and reasons does not define which rules are valid, and which reasons exist. In our view, such facts can only be determined by means of empirical investigation: which reasons exist and which rules are valid in a given reasoning community is shown by the argumentation behavior of the reasoners in that community. Which rules are valid and which reasons exist is not determined by logic.

Just as with other empirically studied domains it cannot be expected that the empirical data lead to a unique and indisputable theory of rule validity. Moreover, it will often happen that new data gives rise to a revision of the theory of rule validity. To complicate matters further, rule validity can change with time and there is not always general agreement about rule validity in a community.[9]

[9] Even in the mathematical community where argumentation takes the form of mathematical proof the ideas about rule validity can change and can be the subject of

Therefore, examples are always based on a theory of rules and reasons that is given beforehand as a set of premises.

To summarize, the goal of this chapter is twofold:

- We specify types of facts concerning rules and reasons by defining an appropriate language containing sentences that express these types of facts (section 4).
- We specify the relations that must hold between the types of facts, in terms of the relations between the truth values of sentences in this language (section 5).

After the formal description of the semantics of rules and reasons, we discuss which conclusions follow from given premises (section 6). In agreement with our method (chapter 1, section 7), we continue with an introduction of the formalism, by means of the examples from section 1.

3 Towards a formalization

In the examples of section 1, we have encountered several types of facts concerning rules and reasons. For instance, a rule can be valid, it can be applied, and reasons can be weighed. Reason-Based Logic is a formalism in which such facts can be formally represented, and that makes the relations between these facts precise. In this section, we use the informal examples of section 1 to introduce this formalism.

3.1 Rules and reasons

The conclusion of an argument is supported by reasons. For instance, in the argument

> According to the weather report, it will rain all day.
> So, it will rain all day.

'According to the weather report, it will rain all day' is a reason for 'It will rain all day'.[10] As the language of Reason-Based Logic, we use the language of First-

dispute. Examples are Brouwer's constructivist view on mathematical proof, and more recently the dispute about the acceptability of computers as proof tools (cf. Stewart, 1996).

[10] Actually, we should say that the *state of affairs* expressed by the sentence 'According to the weather report, it will rain all day' is a reason for *state of affairs* expressed by the sentence 'It will rain all day'. (Cf. the difference between states of affairs and the sentences expressing them discussed in section 2.) For convenience, however, we will not use this extensive expression.

Order Predicate Logic.[11] A number of special function and predicate symbols are used to express the notions that are typical for Reason-Based Logic. The premise and the conclusion of the argument above can be represented as Weather_report(rainy_day) and Rainy_day, respectively.

In the argument, Weather_report(rainy_day) is a reason for Rainy_day.[12] In Reason-Based Logic, a special predicate is used to express this fact:

```
Reason(weather_report(rainy_day),
    rainy_day)
```

This sentence expresses a state of affairs that some state of affairs is a reason for another. As a result, the sentence, expressing one state of affairs, contains references to other states of affairs. Here we encounter an important subtlety in the language of Reason-Based Logic: states of affairs are expressed by sentences of the language, and referred to by terms in other sentences. For instance, the state of affairs that, according to the weather report, it will rain all day, is expressed by the sentence

```
Weather_report(rainy_day)
```

and referred to by the term

```
weather_report(rainy_day)
```

in the sentence

```
Reason(weather_report(rainy_day),
    rainy_day).
```

As a result, in the language of Reason-Based Logic, there is a translation from sentences to terms. In order to distinguish between sentences expressing states of affairs and terms referring to them, a typographical convention is used: a string with an initial upper-case character is a sentence, and a string with an initial lower-case character a term (see section 4.3 for details).

We have discussed that reasons do not arise individually, but follow a pattern (section 1.1). The reason above instantiates the following reason scheme:

---

[11] For an introduction to First-Order Predicate Logic, see Van Dalen (1983) or Davis (1993).

[12] It may seem sloppy that we use the same phrase '... is a reason for ...' in the sentence ''According to the weather report, it will rain all day' is a reason for 'It will rain all day'' and in the sentence 'Weather_report(rainy_day) is a reason for Rainy_day'. However, no confusion can arise, since both 'According to the weather report, it will rain all day' and Weather_report(rainy_day) express the same state of affairs, only in different language.

Reason(weather_report(*weather_type*),
weather_type)

Here, *weather_type* is a variable, representing some type of weather.[13] The reasons matching this pattern arise by the application of a valid rule. The rule can be represented as follows

rule(weather_report(*weather_type*),
weather_type)

The rule has a condition weather_report(*weather_type*) and a conclusion *weather_type*. The use of lower-case characters shows that the rule is represented as a term: we treat a rule as an object that represents a relation between condition and conclusion. The fact that this rule is valid is expressed by the following sentence:

Valid(rule(weather_report(*weather_type*),
weather_type)) (1)

The rule gives rise to a reason if it applies. In our example, the rule applies *(initially) since Weather_report(rainy_day) is true. The fact that the rule above* applies is expressed as

Applies(rule(weather_report(*weather_type*),
weather_type),
weather_report(rainy_day),
rainy_day).

This sentence expresses that the rule with condition weather_report(*weather_type*) and conclusion *weather_type* applies on the basis of the fact weather_report(rainy_day).

3.2 Exclusionary reasons

A rule normally applies if its condition is satisfied. However, as we have seen, this is not always the case. For instance, a rule does not apply if the rule is excluded because of an exclusionary reason. We saw that 'The weather report on national television is not good at predicting the local weather' was an exclusionary reason against the applicability of the rule (1) above. This fact is expressed by the following sentence:

---

[13] This suggests that a formal language with typed variables could be useful (see for instance Davis, 1993, p. 40ff. on many-sorted logic). We will not do this, in order to make the formalism not unnecessarily complicated.

```
Reason(bad_local_prediction,
    excluded(rule(weather_report(weather_type),
            weather_type),
        weather_report(rainy_day),
        rainy_day))
```

As a result, the rule (1) is excluded:

```
Excluded(rule(weather_report(weather_type),
        weather_type),
    weather_report(rainy_day),
    rainy_day)
```

Since the rule is excluded, the rule is not applicable. In Reason-Based Logic, this is expressed as follows:

```
¬Applicable(rule(weather_report(weather_type),
        weather_type),
    weather_report(rainy_day),
    rainy_day)
```

In the example, the rule does not apply, and the sentence

```
Reason(weather_report(rainy_day),
    rainy_day)
```

is false. So far, there is no reason to conclude that it will rain.

3.3 Weighing reasons

Later in our story John and Mary had two reasons concerning the weather that Sunday:

```
Reason(belief_father(rainy_day),
    rainy_day)
Reason(cloudless_sky,
    ¬rainy_day)
```

So, Belief_father(rainy_day) is a reason for Rainy_day, while Cloudless_sky is a reason for ¬Rainy_day, i.e., a reason against Rainy_day. In such a case of conflicting reasons, the reasons must be weighed. John and Mary decide that Cloudless_sky as a reason against Rainy_day outweighs the reason Belief_father(rainy_day). This is expressed as:

Outweighs({cloudless_sky},
{belief_father(rainy_day)},
¬rainy_day)

More precisely, this sentence expresses that the set of reasons containing only the reason Cloudless_sky for ¬rainy_day (i.e, against rainy_day) outweighs the set of reasons containing only the reason Belief_father(rainy_day) for rainy_day. Sets of reasons are needed since there can be several reasons pointing in the same direction.

3.4 Reasons concerning the application of a rule

We saw that there can be reasons for and against the application of a rule. In our example, John and Mary knew that if they would apply the rule (1) and as a result conclude that it will rain, their trip would no longer be enjoyable. That gives a reason against the application of the rule:

Reason(trip_no_longer_enjoyable,
¬applies(rule(weather_report(*weather_type*),
weather_type),
weather_report(rainy_day),
rainy_day))

However, the condition of the rule is satisfied, since Weather_report(rainy_day) is true (after John and Mary hear the radio). The rule is this time not excluded, so it is applicable. If a rule is applicable, the fact that makes it applicable is a reason to apply the rule. So, there is also a reason for applying the rule:

Reason(weather_report(rainy_day),
applies(rule(weather_report(*weather_type*),
weather_type),
weather_report(rainy_day),
rainy_day))

John and Mary consider the reason not to apply the rule stronger:

Outweighs({trip_no_longer_enjoyable},
{weather_report(rainy_day)},
¬applies(rule(weather_report(*weather_type*),
weather_type),
weather_report(rainy_day),
rainy_day))

and they do not apply the rule.

4 Types of facts

In this section, we start with the formal definition of Reason-Based Logic.[14] We specify the types of facts concerning rules and reasons by defining a formal language in which the different types of facts can be expressed.

The language of Reason-Based Logic (RBL) is based on that of First-Order Predicate Logic (FOPL).[15] However, there are differences since the language of Reason-Based Logic must be appropriate to represent the types of facts concerning rules and reasons that we have encountered.

The main differences are that the language of Reason-Based Logic contains a number of special function and predicate symbols, and that there is a translation from sentences to terms.

As a result, terms and sentences must adhere to certain constraints. Therefore, after the definition of alphabets (section 4.1), we must distinguish between pre-terms and pre-sentences, not adhering to the constraints, and terms and sentences, adhering to the constraints. In section 4.2, pre-terms and pre-sentences are defined, analogous to terms and sentences of First-Order Predicate Logic. In section 4.3, we define the translation from sentences to terms. In section 4.4, we then define terms and sentences as pre-terms and pre-sentences adhering to certain constraints. Section 4.5 contains an overview of the types of facts.

4.1 Alphabets of Reason-Based Logic

The following definition shows that an alphabet of Reason-Based Logic is identical to an alphabet of First-Order Predicate Logic that contains some special-purpose function and predicate symbols.

Definition 1.

Function symbols are finite strings of symbols a, b, c, ..., z, A, B, C, ..., Z, _ starting with a lower-case.

Predicate symbols are finite strings of symbols a, b, c, ..., z, A, B, C, ..., Z, _ starting with an upper-case.

Variable symbols are finite strings of symbols *a*, *b*, *c*, ..., *z*, *A*, *B*, *C*, ..., *Z*, _ starting with a lower-case.

An *alphabet* of Reason-Based Logic is any set consisting of

---

[14] Several versions of Reason-Based Logic have been presented over the years, e.g., by Hage (1991, 1993, 1995), Hage and Verheij (1994a, b) and Verheij (1993, 1994, 1995e). The differences are mainly due to new insights or differences of focus. For instance, Hage (1995) has extended Reason-Based Logic to incorporate reasoning with goals, while Verheij (1994) used a limited version of Reason-Based Logic to get the formalism right. See also note 1.

[15] In the following we do not go into details of First-Order Predicate Logic, and assume that the reader has some familiarity with it. For instance, Van Dalen (1983) gives a good introduction to the syntax and semantics of First-Order Predicate Logic.

1. the function symbol rule with arity 2, plus any number of additional function symbols, each assigned a natural number denoting its arity,
2. the predicate symbols Reason with arity 2, Valid with arity 1, Excluded with arity 3, Applicable with arity 3, Applies with arity 3, and Outweighs with arity 3, plus any number of additional predicate symbols, each assigned a natural number denoting its arity,
3. variable symbols, and
4. the symbols (,), {, }, ¬, ∧, ∨, →, ∃, ∀, =, : and ,.[16]

Function and predicate symbols do not need to have a unique arity.

The smallest alphabet consists of the function symbol rule with arity 2, predicate symbols Reason with arity 2, Valid with arity 1, Excluded with arity 3, Applicable with arity 3, Applies with arity 3, and Outweighs with arity 3, no variable symbols, and the symbols (,), {, }, ¬, ∧, ∨, →, ∃, ∀, =, : and ,. The largest alphabet consists of all function predicate symbols (with all arities), all variable symbols, and the symbols (,), {, }, ¬, ∧, ∨, →, ∃, ∀, =, : and ,.

In the following, the definitions refer to a fixed alphabet of Reason-Based Logic.

4.2 Pre-terms and pre-sentences

Before we can define the terms and sentences of Reason-Based Logic, we need to define pre-terms and pre-sentences. These are defined in a similar way as the terms and sentences of First-Order Predicate Logic. (The terms and sentences of Reason-Based Logic have to adhere to certain additional constraints.) In the following definition, n denotes a natural number, $n > 0$, except when otherwise indicated.

Definition 2.

The set of *pre-terms* of Reason-Based Logic is the smallest set such that the following holds:

1. Any function symbol with arity 0 and any variable symbol is a pre-term.
2. If $term_1$, $term_2$, ..., and $term_n$ are pre-terms and *function* is a function symbol with arity n, then $function(term_1, term_2, ... term_n)$ is a pre-term.
3. If $term_1$, $term_2$, ..., and $term_n$, with $n \geq 0$, are pre-terms, then $\neg term_1$, $(term_1 \wedge term_2)$, $(term_1 \vee term_2)$ and $\{term_1, term_2, ..., term_n\}$ are pre-terms.

For convenience, we use the same typographical style for variable symbols and metavariables. The role of the pre-terms of the forms $\neg term_1$, $(term_1 \wedge term_2)$, $(term_1 \vee term_2)$ and $\{term_1, term_2, ..., term_n\}$ will be explained below (section 4.3).

Three examples of pre-terms are:

[16] Here the comma ',' (of the normal text font) is used to separate the symbols of the alphabet, and the comma ',' (of the formula font) is one of the symbols.

mary
father(john)
rule(weather_report(*weather_type*),
weather_type)

Definition 3.

The set of *pre-formulas* is the smallest set such that the following hold:

1. If $term_1$ and $term_2$ are pre-terms, then $term_1 = term_2$ is a pre-formula.
2. Any predicate symbol with arity 0 is a pre-formula.
3. If $term_1$, $term_2$, ..., and $term_n$ are pre-terms and *Predicate* is a predicate symbol with arity n, then $Predicate(term_1, term_2, \ldots term_n)$ is a pre-formula.
4. If $Formula_1$ and $Formula_2$ are pre-formulas, then $\neg Formula_1$, $(Formula_1 \wedge Formula_2)$, $(Formula_1 \vee Formula_2)$ and $(Formula_1 \rightarrow Formula_2)$ are pre-formulas.
5. If *Formula* is a pre-formula and *x* is a variable symbol, then $\exists x$: *Formula* and $\forall x$: *Formula* are pre-formulas.

A *pre-atom* is a pre-formula of one the forms *Predicate*, $term_1 = term_2$, or $Predicate(term_1, term_2, \ldots term_n)$. A *pre-literal* is a pre-atom or a pre-atom preceded by $\neg$. A *pre-sentence* is a pre-formula without free variables.[17]

We use the ordinary conventions to reduce the number of brackets in formulas.

Three examples of pre-sentences and pre-formulas are:

Is_thief(mary)
Predicts(local_weather_report, rainy_day)
Valid(rule(weather_report(*weather_type*),
weather_type))

Shortly, we will see that not all pre-terms and pre-sentences are terms and sentences of Reason-Based Logic. We return to this issue in section 4.4.

4.3 A translation from sentences to terms

As mentioned in section 3.1, in Reason-Based Logic, we do not only need to express states of affairs as sentences, but also to refer to them in other sentences. In the formal language, we use a translation from (pre-)sentences to (pre-)terms in order to refer to sentences.[18]

We use a simple translation: to obtain the pre-term that corresponds to a (quantifier free) pre-sentence, the first upper-case character of each predicate

[17] Free variables are defined as usual.

[18] This is an often-encountered technique, known as *reification*. For other examples, we refer to the overview of meta-languages, reflection principles and self-reference by Perlis and Subrahmanian (1994).

symbol in the pre-sentence is replaced by the same character in lower-case. By the choice of alphabet and the definition of terms, the result of this translation is always a pre-term.

For example, the pre-sentence

Is_thief(mary)

translates to the term

is_thief(mary).

As the definition of pre-terms shows, the logical connectives are treated as if they also are function symbols. In this way, the translation can be kept as simple as it is now. For example, the pre-sentence

Is_guilty(mary) ∧ ¬Punish(mary)

translates to the term

is_guilty(mary) ∧ ¬punish(mary).

To stay as close as possible to the usual notation of sentences, the logical connectives are *infix* function symbols. For instance, instead of writing terms of the form $\wedge(term_1, term_2)$, we write $term_1 \wedge term_2$.

Of course not all terms should be translations of sentences. For instance, the terms mary and father(john) do not correspond to sentences Mary and Father(john). Therefore, we should divide the set of terms into two types, namely those that correspond to sentences, and those that do not. As a result, only a subset of all strings of characters beginning with an upper-case can be predicate symbols. For convenience, we will not explicitly define such a subset, but assume that any string of characters beginning with an upper-case that we encounter is in this subset.

The translation easily extends to metavariables for (pre-)sentences and (pre-)terms, as follows. Metavariables for pre-sentences will be denoted as strings of italic characters beginning with an upper-case character, e.g., *Fact*. Metavariables for pre-terms will be denoted as strings of italic characters beginning with a lower-case (just as the variables of the logical language), e.g., *fact*. Matching metavariables for pre-sentences and pre-terms, such as *Fact* and *fact*, represent a sentence and its translation to a term. This extended translation will turn out to be crucial in several of the coming definitions.

4.4 Terms and sentences

In Reason-Based Logic, there are function and predicate symbols that play a special role. There are constraints on their use. Formally, we define terms and formulas as pre-terms and pre-formulas that adhere to a number of constraints.

Definition 4.

A *term* of Reason-Based Logic is a pre-term that adheres to the following constraints:

1. If rule(*condition*, *conclusion*) is a pre-term, *Condition* must be a disjunction of conjunctions of pre-literals and *Conclusion* a pre-literal.
2. If {$fact_1$, $fact_2$, ..., $fact_n$} is a pre-term, $Fact_1$, $Fact_2$, ..., and $Fact_n$ must be disjunctions of conjunctions of pre-literals.

In this definition, we use the translation from sentences to terms for the first time in a formal definition. For instance, *Condition* denotes a sentence that translates to the term denoted by *condition*.

Definition 5.

A *formula* of Reason-Based Logic is a pre-formula *Formula* that adheres to the following constraints:

1. All pre-terms that occur in *Formula* must be terms.
2. If *Formula* has the form

   Reason(*fact*, *state_of_affairs*),
   Valid(rule(*condition*, *conclusion*)),
   Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*),
   Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*),
   Applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*), or
   Outweighs($reasons_1$, $reasons_2$, *state_of_affairs*),

   then the following must hold:
   a. *Fact*, *State_of_affairs*, $Reason_1$, $Reason_2$, ... and $Reason_n$ must be pre-sentences, i.e., do not contain free variables.
   b. *Fact* must be a disjunction of conjunctions of pre-literals and must be an instance of *Condition* under some substitution σ, and *State_of_affairs* must be a pre-literal that is an instance of *Conclusion* under the same substitution σ.
   c. The (pre-)terms $reasons_1$ and $reasons_2$ must both have the form {$fact_1$, $fact_2$, ..., $fact_n$}, with $n \geq 0$.

Atoms and *literals* are formulas that are pre-atoms and pre-literals, respectively. *Sentences* are pre-formulas that only contain free variables in occurrences of terms of the form rule(*condition*, *conclusion*).

Definition 6.

A *language* of Reason-Based Logic is the set of formulas belonging to some alphabet of Reason-Based Logic.

4.5 Overview of the types of facts

As we saw in section 3, in Reason-Based Logic, a number of function and predicate symbols are used to express types of facts concerning rules and reasons. Below we provide an overview of these function and predicate symbols and their use.

- rule(*condition*, *conclusion*)

  Since we treat rules as objects, rules are represented as terms in Reason-Based Logic. In this way it is possible to express facts about rules. A term denoting a rule has the form rule(*condition*, *conclusion*). Here *condition* and *conclusion* are terms with free variables. The formula *Condition* that translates to the term *condition* must be a disjunction of conjunctions of one or more literals. In other words, *Condition* is quantifier free and in disjunctive normal form. An instance of *Condition* is a possible reason for a matching instance of *Conclusion*. The formula *Conclusion* that translates to the term *conclusion* must be a literal.

- {$fact_1$, $fact_2$, ..., $fact_n$} (for n = 1, 2, ...)

  These symbols are used to refer to the sets of facts that are reasons for some conclusion. We use an unusual syntax of terms to stay as close as possible to the normal notation of sets. The term {thief(mary), minor(mary)} refers to the set of the two reasons expressed by the sentences Thief(mary) and Minor(mary). The term { } (without arguments) is used to denote an empty set of reasons.

  There is a problem here with different terms that denote identical sets, such as {thief(mary), minor(mary)} and {minor(mary), thief(mary)}. Axioms should be included in Reason-Based Logic such that formulas that only differ in such equivalent terms for sets are equivalent. We will not do this explicitly.

  We do not consider infinite sets of reasons.

- Reason(*fact*, *state_of_affairs*)

  A sentence of this form expresses that the fact referred to by the term *fact* is a reason for the state of affairs referred to by the term *state_of_affairs*. The sentence *Fact* (that translates to the term *fact*) must be a disjunction of conjunctions of literals, and *State_of_affairs* (that translates to the term *fact*) a literal. If *State_of_affairs* is an atom *Atom*, *Fact* is a reason for *Atom* and a reason against ¬*Atom*; similarly, if *State_of_affairs* is a negated atom ¬*Atom*, *Fact* is a reason for ¬*Atom* and a reason against *Atom*.

- Valid(rule(*condition*, *conclusion*))

  A sentence of this form expresses that the rule with condition *condition* and conclusion *conclusion* is valid.

- Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)

  A sentence of this form expresses that the rule with condition *condition* and conclusion *conclusion* is excluded, for the instance *Fact* of the rule's condition *Condition*. *Fact* must be an instance of *Condition*, and *State_of_affairs* an instance of *Conclusion*.

- Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)

  A sentence of this form expresses that the rule with condition *condition* and conclusion *conclusion* is made applicable by the fact expressed by the term *fact*. If a rule is applicable, it may give rise to a reason for the state of affairs expressed by the term *state_of_affairs*. *Fact* must be an instance of one of the disjuncts of *Condition*, and *State_of_affairs* an instance of *Conclusion*.

- Applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)

  A sentence of this form expresses that the rule with condition *condition* and conclusion *conclusion* applies on the basis of the fact expressed by *fact* and therefore generates a reason for the state of affairs expressed by *state_of_affairs*. *Fact* must be an instance of *Condition*, and *State_of_affairs* an instance of *Conclusion*. The predicate *Applies* should not be confused with the predicate *Applicable*. The difference in meaning (introduced in the sections 1 and 3) is made precise in the next section.

- Outweighs(*reason_pro*, *reasons_con*, *state_of_affairs*)

  A sentence of this form expresses that the reasons in the set referred to by the term *reasons_pro* outweigh the reasons in the set referred to by the term *reasons_con* (as reasons concerning *state_of_affairs*). The terms *reasons_pro* and *reasons_con* must both have the form $\{fact_1, fact_2, ..., fact_n\}$, where $n \geq 0$. Each sentence $Fact_i$ must be a disjunction of conjunctions of literals (for each i from 1 to n), and *State_of_affairs* a literal. The reasons in *reasons_pro* are reasons for *State_of_affairs*, and the reasons in *reasons_con* are reasons against *State_of_affairs*. Equivalently, if *Not_state_of_affairs* is the literal that is the opposite of *State_of_affairs*, the reasons in *reasons_pro* are reasons against *Not_state_of_affairs*, and the reasons in *reasons_con* are reasons for *Not_state_of_affairs*.

5 Relations between facts

In this section, we describe the relations that hold between the described facts concerning rules and reasons. We do it in terms of the truth values of the corresponding sentences. The basis is again First-Order Predicate Logic.[19] The relations that hold between facts (in terms of the truth values of sentences that express the facts) as defined by First-Order Predicate Logic also hold in Reason-Based Logic. For instance, the following relations hold:

NOT
For all sentences *State_of_affairs*,
Either *State_of_affairs* is true or ¬*State_of_affairs* is true.

AND
For all sentences $State\_of\_affairs_1$ and $State\_of\_affairs_2$,
$State\_of\_affairs_1$ is true and $State\_of\_affairs_2$ is true if and only if
$State\_of\_affairs_1 \wedge State\_of\_affairs_2$ is true.

OR
For all sentences $State\_of\_affairs_1$ and $State\_of\_affairs_2$,
$State\_of\_affairs_1$ is true or $State\_of\_affairs_2$ is true if and only if
$State\_of\_affairs_1 \vee State\_of\_affairs_2$ is true.

The relations that hold between sentences that are typical for Reason-Based Logic are defined in a similar way. They are called VALIDITY, EXCLUSION, APPLICABILITY, APPLICATION, WEIGHING, and WEIGHING_AXIOMS.[20] We assume in the following that all mentioned sentences are well-formed, i.e., are sentences of the language of Reason-Based Logic.

VALIDITY
For all sentences *Condition*, *Conclusion*, *Fact* and *State_of_affairs*,
If Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*),
Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) or
Applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is true, then
Valid(rule(*condition*, *conclusion*)) is true.

---

[19] For convenience, we will not as usual define the relations between facts in terms of structures and models, but in terms of truth values of sentences. Such a definition can be given, but does not provide additional insight, while the formalism becomes more complex.

[20] These relations could also be given as a set of axioms. We have chosen the present form in order to stress that in Reason-Based Logic the standard logical connectives, such as ¬ and ∧, are not treated differently from the non-standard logical constants, such as Valid and Applicable.

Informally, VALIDITY says that a rule can only be excluded, be applicable, or apply if it is valid.

EXCLUSION

For all sentences *Fact* and *State_of_affairs*,

If *Fact* and Valid(rule(*condition*, *conclusion*)) are true, then either Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) or Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is true.

Informally, EXCLUSION says that a rule is either excluded or applicable if its condition is satisfied. Here *Fact* stands for the fact that satisfies the condition of the rule.

APPLICABILITY

For all sentences *Fact* and *State_of_affairs*,

a. Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is true if and only if Reason(*fact*, applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)) is true.
b. If Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is true, then *Fact* is true.

Informally the first part of APPLICABILITY says that if and only if a rule is applicable, the fact that makes the rule applicable is a reason to apply the rule. The second part says that a rule can only be applicable if its condition is satisfied. Again, *Fact* stands for the fact that satisfies the condition of the rule.

APPLICATION

For all sentences *Fact* and *State_of_affairs*,

There are terms *condition* and *conclusion*, such that Applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is true if and only if Reason(*fact*, *state_of_affairs*) is true.

Informally this relation says that if and only if a rule applies, the fact that makes the rule applicable is a reason for the rule's (instantiated) conclusion. or, equivalently, a reason against the opposite of the rule's conclusion.

Notice the difference between a rule's being applicable and its being applied. If a rule is applicable, this only indicates that there is a reason for applying the rule (see APPLICABILITY, part a). In general, there can also be reasons against applying a rule.

WEIGHING

For all sentences Pro_1, Pro_2, ..., Pro_n (for some natural number n), Con_1, Con_2, ..., Con_m (for some natural number m), *State_of_affairs*, and its opposite *Not_state_of_affairs*,

If Reason(pro_1, *state_of_affairs*), Reason(pro_2, *state_of_affairs*), ..., Reason(pro_n, *state_of_affairs*), Reason(con_1, *not_state_of_affairs*), Reason(con_2, *not_state_of_affairs*), ..., Reason(con_m, *not_state_of_affairs*), and also Outweighs({pro_1, pro_2, ..., pro_n}, {con_1, con_2, ..., con_m}, *state_of_affairs*) is true, then *State_of_affairs* is true, or there is a term *con*, different from con_1, con_2, ..., and con_m, such that Reason(*con*, *not_state_of_affairs*) is true.

Informally the first part of this relation says that reasons make a conclusion true if the pros outweigh the cons, provided that no con is overlooked. It is allowed that one or more of the pros is overlooked: if a subset of the pros already suffices to outweigh all cons, the conclusion certainly follows if there are even more pros.[21] It may seem that a similar relation between facts is required for the case that the cons outweigh the pros. However, since in Reason-Based Logic a reason against a state of affairs is just a reason for the opposite state of affairs, the relation above suffices.[22]

WEIGHING_AXIOMS

For all sentences $Fact_1$, $Fact_2$, ..., $Fact_n$ (for some positive natural number n), *State_of_affairs*, and its opposite *Not_state_of_affairs*, and all terms *pros* and *cons*,

a. Outweighs(*pros*, *cons*, *state_of_affairs*) and Outweighs(*cons*, *pros*, *not_state_of_affairs*) are not both true.
b. If Reason($fact_1$, *state_of_affairs*), Reason($fact_2$, *state_of_affairs*), ..., Reason($fact_n$, *state_of_affairs*) are true, then Outweighs({$fact_1$, $fact_2$, ..., $fact_n$}, { }, *state_of_affairs*) is true.

The first part of this relation says that the pros as reasons for *state_of_affairs* cannot outweigh the cons and the other way around at the same time. However, the first weighing axiom does not make it impossible that ¬Outweighs(*pros*, *cons*, *state_of_affairs*) and ¬Outweighs(*cons*, *pros*, *state_of_affairs*) are both true.

Reason-Based Logic does in general not determine which set of reasons outweighs another set. However, for the case that all reasons point in the same direction, i.e., all reasons are either pros or cons, the second part of the relation gives the result: any non-empty set of reasons outweighs the empty one.

---

21 This is due to the *accrual of reasons*, a term used by Pollock (1991, p. 51). Accrual is discussed more extensively later on.

22 In other versions of Reason-Based Logic (e.g., Hage and Verheij, 1994a), the two cases that the pros outweigh the cons and that the cons outweigh the pros, are formally distinguished, even though there is no conceptual distinction. In the version of Reason-Based Logic described by Verheij (1995e), this is acknowledged, and the two cases are no longer formally distinguished.

6 Conclusions following from given premises

Although it is not strictly part of the semantics of rules and reasons, we discuss in this section which conclusions follow from given premises. The given set of premises, representing a theory of rules and reasons, is called a theory of Reason-Based Logic.

The simplest approach is to define which conclusions deductively follow from a given theory analogous to First-Order Predicate Logic, as follows:

Definition 7. (RBL-deduction)

A *theory* of Reason-Based Logic is any set of sentences (in a given language of Reason-Based Logic). A conclusion *Conclusion deductively follows* from a theory T, if the truth of the sentences in T follows from the truth of the sentence *Conclusion*, using the relations between facts of Reason-Based Logic.[23]

Definition 7 extends deduction in First-Order Predicate Logic, and allows that conclusions are drawn on the basis of the relations between facts that hold in Reason-Based Logic. It is possible to define a set of deduction rules, in the style of First-Order Predicate Logic's natural deduction, that are sound and complete with respect to this deductive consequence relation. However, this consequence relation turns out to be weak, and intuitively attractive types of reasoning on the basis of reasons are not captured by RBL-deduction.

As a result, we do not devote much attention to the deductive consequence relation, and focus on a more interesting *nonmonotonic* consequence relation.

We give an example of a type of reasoning that is not captured by RBL-deduction: the conclusion Rainy_day does not follow from the theory that consists of the two sentences

Weather_report(rainy_day)
Valid(rule(weather_report(*weather_type*),
weather_type))

Intuitively, simply applying the rule, the condition of which is satisfied, leads to the conclusion Rainy_day. The difficulty is hidden in the world 'simply': the rule does not simply apply, since for the rule to apply several semantical constraints must be met. As a result, not in all circumstances in which the theory above is true, the rule actually applies. Formally, the sentence

Applies(rule(weather_report(*weather_type*),
weather_type),

[23] Normally, which conclusions follow from a theory is defined in terms of the *models* of the theory. But see note 19.

weather_report(rainy_day),
rainy_day)

is not always true. For instance, it can be the case that the rule is excluded, i.e., in which

Excluded(rule(weather_report(*weather_type*),
weather_type),
weather_report(rainy_day),
rainy_day)

is true. Then the rule is not applicable and normally not applied.

Intuitively, however, it seems most natural that the rule is not excluded, since there is no information in the theory that makes it excluded. Therefore, it seems natural to allow the following type of reasoning:

> If the condition of a rule is satisfied, then it follows that the rule is applicable, unless it follows that the rule is excluded.

This type of reasoning is an example of a *nonmonotonic* rule of inference. It is called nonmonotonic, since it can be the case that conclusions based on it must be retracted because of newly inferred facts. For instance, it may seem now that a rule is not excluded with respect to the currently inferred facts, but later it may be inferred that the rule is excluded after all. This is in contrast with the usual monotonic rules of inference. Once a conclusion based on monotonic rules of inference is established, it never has to be retracted on the basis of newly inferred facts.

The problem with nonmonotonic rules of inference is that they can only be safely used to draw conclusions if one knows all consequences of a theory in advance. This is in conflict with the step by step construction of the set of consequences of a theory: starting from the premises in the theory conclusions are added step by step by drawing new conclusions using the rules of inference. As a result, the complete set of conclusion following from a theory is only known after all steps have been completed.

Many approaches to deal with nonmonotonic rules of inference have been proposed. Ginsberg (1987), Lukaszewicz (1990) and Gabbay *et al.* (1994b) have given overviews of such research. We present an approach based on extensions that is related to ideas that go back to Reiter's Default Logic (1980, 1987).

In the definition of the nonmonotonic consequences of a theory we use a set of sentences that can be regarded as a guess in advance of the set of consequences. The nonmonotonic rule of inference mentioned above is then read in a slightly different way, by referring to this guess:

If the condition of a rule is satisfied, then it follows that the rule is applicable, unless it is guessed that it follows that the rule is excluded.

Let now T be a theory, and S a set of sentences, that represents our guess of consequences following from the theory T. We will define which conclusions follow from the theory T relative to the guess set S. The following rule of inference (related to the relation between facts EXCLUSION of section 5) holds in Reason-Based Logic:

EXCLUSION*

For all sentences *Fact* and *State_of_affairs*,

If *Fact* and Valid(rule(*condition*, *conclusion*)) follow from T relative to S, then Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) follows from T relative to S, unless Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) is an element of S.

This rule of inference says that if it follows that the condition of a rule is satisfied, it follows that the rule is applicable, unless it is guessed that the rule is excluded.

There is a second type of reasoning that is intuitively attractive, but is not captured by RBL-deduction. Informally:

If it follows that all derivable pros outweigh all derivable cons, the conclusion of the pros follows. If it follows that all derivable cons outweigh all derivable pros, the conclusion of the cons follows.

This type of reasoning is however also an example of a nonmonotonic rule of inference. Since it refers to all derivable pros and cons, one has to know the whole set of conclusions in advance. Again we use the fixed guess set S to avoid the difficulties. Instead of using all derivable pros and cons, the following rule of inference (related to the relation between facts WEIGHING of section 5) uses all reasons in the guess set S.

WEIGHING*

For all sentences Pro_1, Pro_2, ..., Pro_n (for some natural number n), Con_1, Con_2, ..., Con_m (for some natural number m), *State_of_affairs*, and its opposite *Not_state_of_affairs*,

If Reason(pro_1, *state_of_affairs*), Reason(pro_2, *state_of_affairs*), ..., Reason(pro_n, *state_of_affairs*), Reason(con_1, *not_state_of_affairs*), Reason(con_2, *not_state_of_affairs*), ..., Reason(con_m, *not_state_of_affairs*), and also Outweighs({pro_1, pro_2, ..., pro_n}, {con_1, con_2, ..., con_m}, *state_of_affairs*) follow from T relative to S, then *State_of_affairs* follows from T relative to S, unless there is a term *con*, different from con_1, con_2, ..., and con_m, such that Reason(*con*, *not_state_of_affairs*) is an element of S.

Finally, the conclusions that deductively follow from a theory also follow from T relative to S:

RBL-DEDUCTION
1. All elements of T follow from T relative to S.
2. All sentences that (deductively) follow from sentences that follow from T relative to S follow from T relative to S.

The conclusions that follow from a theory T relative to a guess set S can now be defined as in First-Order Predicate Logic by a recursive definition using the rules of inference EXCLUSION*, WEIGHING* and RBL-DEDUCTION. The problems of such a recursive definition for nonmonotonic rules of inference have been avoided by translating these rules to monotonic rules of inference relative to the fixed set S. We have the following ordinary recursive definition of the conclusions that follow from a theory relative to a guess set:

Definition 8. (S-consequences)
A *guess set* of Reason-Based Logic is any set of sentences (in a given language of Reason-Based Logic). For any theory T and any guess set S, the set of conclusions that *follow from* T *relative to* S is the smallest set of sentences, such that EXCLUSION*, WEIGHING* and RBL-DEDUCTION hold. The conclusions that follow from T relative to the guess set S, are the S-*consequences* of T.

If T is a theory and S is a guess set, there are two cases in which the guess set S is not acceptable as a set of nonmonotonic consequences of T. First the guess set can be too small: there are S-consequences of T that are not in the guess set S. Second the guess set can be too large: not all sentences in the guess set S are S-consequences of T. So, a guess set is a set of nonmonotonic consequences of T if and only if the guess set is equal to the set of consequences relative to the guess set. A set of nonmonotonic consequences is usually called an extension. We get the following fixed-point definition:

Definition 9. (extensions)
For any theory T, a set of sentences E is an *extension* if and only if E is equal to the set of E-consequences of T.[24]

[24] One can see that this definition of extension corresponds to Reiter's (1980, 1987) if one reads the rules EXCLUSION* and WEIGHING* as defaults. For instance, EXCLUSION* corresponds to defaults with prerequisite *Fact* $\wedge$ Valid(rule(*condition*, *conclusion*)), justification Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*), and consequent Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*). Our set of S-consequences of a theory T corresponds to Reiter's set $\Gamma_T(S)$. Of course, several unessential technical adaptations are necessary, such as using an RBL language.

A theory does not necessarily have an extension, and, if it has one, the extension is not necessarily unique. For instance, the theory that consists of the sentence

Valid(true, excluded(*condition*, *conclusion*))

has no extension. The theory that consists of the four sentences

A
B
Valid(rule(a, excluded(rule(b, *conclusion*))))
Valid(rule(b, excluded(rule(a, *conclusion*))))

has two extensions, namely one in which the first rule is excluded and the second rule applies, the other in which the first rule applies and the second rule is excluded.

Theories that have no or several extensions contain a paradox resembling the well-known paradoxes of self-reference. In Reason-Based Logic, such paradoxes are possible because of the translation from sentences to terms (as defined in section 4.3). We consider it the task of theories in Reason-Based Logic, rather than of the consequence relation of Reason-Based Logic, to avoid these paradoxes.

Chapter 3

Reason-Based Logic and law

This chapter contains examples of Reason-Based Logic, taken from the field of law. The examples illustrate the basic elements of Reason-Based Logic, and give applications to the theory of legal reasoning.

We start with a discussion of the apparent dichotomy of reasoning with rules and reasoning with principles (section 1). Our claim is that the seeming difference is merely a matter of degree. We support this claim by giving an integrated view on rules and principles (section 2). Before the formal elaboration of this view in Reason-Based Logic (section 7), we discuss how isolated rules/principles, the weighing of reasons, exceptions and conflicts can be modeled in Reason-Based Logic (sections 3, 4, 5 and 6, respectively). We end the chapter with an application of our view on rules and principles to reasoning by analogy (section 8). We show how this view gives rise to three different ways of reconstructing reasoning by analogy.[1]

1 Reasoning with rules vs. reasoning with principles

There seem to be two types of reasoning:

- *Reasoning with rules*
  A rule is applied if its condition is satisfied. If a rule is applied, its conclusion follows directly.

- *Reasoning with principles*
  In contrast with a rule, a principle only gives rise to a reason for its conclusion if it applies. Moreover, there can be other applying principles that give rise to both reasons for and reasons against the conclusion. As a result, a conclusion only follows by weighing the pros and cons.

[1] The sections 1, 3, 7 and 8 of this chapter are based on the papers by Verheij and Hage (1994) and Verheij (1996b).

For instance, Dworkin (1978, p. 22ff. and p. 71ff.) has made a strict distinction between rules and principles in the field of law. According to Dworkin, rules have an all-or-nothing character, while principles have a dimension of weight or importance. An example of a typical rule, he says, is the proposition 'A will is invalid unless signed by three witnesses'. An example of a typical principle is 'No man may profit from his own wrong'.[2]

There are at least three seeming differences between rules and principles. The first is that rules lead directly to their conclusion if they are applied, while principles lead to their conclusion in two steps: first principles give rise to reasons, then these reasons are weighed.

The second difference between rules and principles appears in the case of a conflict. In case of conflicting rules, that is rules with incompatible conclusions that apply to a single case, the rules lead directly to their conclusions, and therefore to a contradiction. In case of conflicting principles, i.e., if there are principles with incompatible conclusions that apply to a single case, no such problems occur. The application of conflicting principles only leads to reasons that plead for incompatible conclusions, so no contradiction is involved. In such cases, a conflict can involve several distinct reasons, some of which plead for a conclusion, others against it. Weighing the pros and cons determines the final conclusion.

The third difference is that rules lead to their conclusion in isolation, while principles interact with other principles. For instance, additional reasons arising from other principles can influence the result of the weighing of reasons.

These differences are summarized in Table 1.

	Rule	**Principle**
Application	Conclusion	Reason
Conflict	Contradiction	Weighing
Other rules/principles	Independent	Dependent

Table 1: The seeming differences between rules and reasons

This leads to the question whether rules and principles are logically different. There is no agreement. For instance, Dworkin has a strong opinion:

> 'The difference between legal principles and legal rules is a logical distinction' (Dworkin, 1978, p. 24)

[2] As Soeteman (1991, p. 33) notes, the usage of the terms 'rule' and 'principle' is not at all uniform. For instance, 'Ne bis in idem' is called a principle, but has a rule-like nature, while 'A contract must be executed in good faith' is a principle-like rule. Here, we do not deal with the usage of the terms 'rule' and 'principle', but with the nature of rules and principles.

Soeteman (1991), in his discussion of rules and principles, takes an apparently opposite stand:

> 'I know of no difference in logical structure between rules and principles.' (Soeteman, 1991, p. 34)[3]

Indeed, there are clear similarities between rules and principles. We mention two of them. First, rules and principles both are basically a connection of some sort between a *condition* and a *conclusion*. The difference is only that, in the case of a rule, the connection seems stronger than in the case of a principle.

Second, for a rule or principle *in isolation* the differences disappear. In isolation, the conclusion of both a rule and a principle follows if the condition is satisfied.

Because of these similarities, we claim that the seeming differences between rules and principles are merely a matter of degree. There is no clear border between reasoning with rules and principles. They are the two extremes of a spectrum.[4] We support our claim by giving an integrated representation of rules and principles in Reason-Based Logic in section 5.[5]

Some preliminaries are required. In the next section we informally discuss our integrated view on rules and principles. Then we discuss how isolated rules/principles, the weighing of reasons, exceptions and conflicts can be represented in Reason-Based Logic (sections 3, 4, 5 and 6, respectively).

2 An integrated view on rules and principles

Our integrated view on rules and principles is based on two main assumptions:

- Both rules and principles give rise to reasons if they are applied.
- The differences between reasoning with rules and principles result from different types of relationships with other rules and principles, which may interfere.

As a basic example of the role of the relationships between rules and principles, we discuss a rule and its underlying principles (section 2.1). Then we discuss our view on a typical rule (section 2.2), a typical principle (section 2.3), and a hybrid rule/principle (section 2.4).

---

3 Translated from the original in Dutch: 'Ik ken (...) geen verschil in logische structuur tussen regels en beginselen'.

4 Soeteman (1991) and Sartor (1994, p. 189) make similar claims. However, our integrated view is more explicit, and can explain the intuitive differences (see section 7).

5 By the formal elaboration, the view can be applied to the use of computers as tools in the field of law (cf. Van den Herik, 1991).

2.1 A rule and its underlying principles

A basic example of the relationships between rules and principles occurs when a rule has underlying principles.

For instance, if the legislator makes a legal rule, this is often based on a decision in which several factors are taken into account. These factors, or to use an already familiar term, reasons, are based on other rules and principles. If these reasons are in conflict, the legislator decides (either explicitly or implicitly) how they have to be weighed. We say that the rules and principles taken into account by the legislator *underlie* the newly made legal rule. In Figure 1, the situation is depicted. The principles underlying the rule that can lead to a reason for the conclusion of the rule are indicated as pro-principles, those that can lead to a reason against the conclusion are indicated as con-principles.

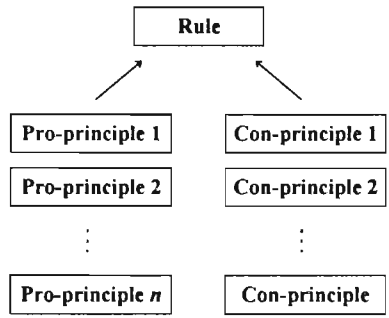

Figure 1: A rule and its underlying principles

As an example, we take the legal rule from Dutch civil law that sale of a house should not terminate an existing rent contract (Art. 7A:1612 BW).[6] This rule has, for instance, the following two underlying principles:

1. Somebody who lives in a house should be protected against measures that threaten the enjoyment of the house
2. Contracts only bind the contracting parties.

The first pleads against termination of an existing rent contract; the second pleads for termination since the new owner of the house does not have a contract with the person (or persons) living in the house. As a result, there is (at least) one underlying pro-principle, and one underlying con-principle.

Let us see what happens if the legal rule applies. Of course, its principles should normally not be applicable too since they have already been considered by

[6] This example is also discussed by Prakken (1993, pp. 22-23) and Verheij and Hage (1994), in the context of analogy. The discussion here is largely taken from the latter. We return to the example in section 8 when dealing with analogy.

the legislator. We say that the legal rule when it applies *replaces* its underlying principles. As a result, if the rule of Art. 7A:1612 BW applies, its two underlying principles should not be applicable. The situation is shown in Figure 2.

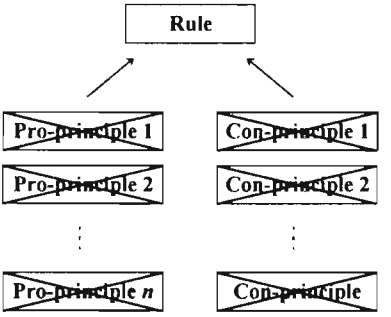

Figure 2: A rule replaces its underlying principles if it applies

If the rule did not replace its underlying principles, several reasons would arise that already had been taken account in the rule itself. However, because of the special relationships of the rule with its underlying principles, the principles should not be applicable.

2.2 A typical rule

In general, the relations between rules and principles are less clear than in the case of a rule and its underlying principles. These relationships can for instance be determined by the weight or importance of a rule or principle, or by the degree of pro- or con-ness. In Figure 3, we have suggested a set of interfering rules and principles.

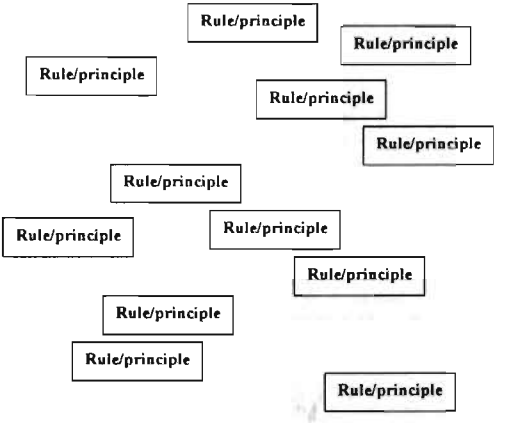

Figure 3: A set of interfering rules and principles

Assume that the rule/principle in the upper left corner is in fact a typical rule. In our view on rules and principles, if this typical rule applies, it blocks all interfering rules/principles. This situation is shown in Figure 4.

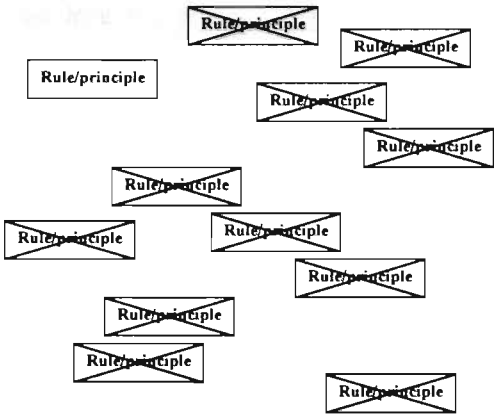

Figure 4: A typical rule applies

As a result, the conclusion of the rule follows directly.

2.3 A typical principle

If the rule/principle in the upper left corner were a typical principle, it would not block any of the interfering rules/principles in case it applies. The situation is shown in Figure 5.

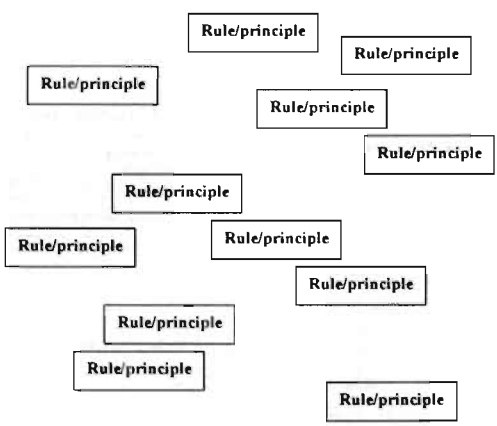

Figure 5: A typical principle applies

As a result, the conclusion of the principle does not follow directly, but only after weighing the reasons arising from the other rules/principles.

2.4 A hybrid rule/principle

Typical rules and typical principles are the extreme cases. Most rules/principles are hybrid: they are neither a typical rule, nor a typical principle. A hybrid rule/principle blocks some, but not all interfering rules/principles. The situation that the rule/principle in the upper left corner were a hybrid rule/principle and applies is shown in Figure 6.

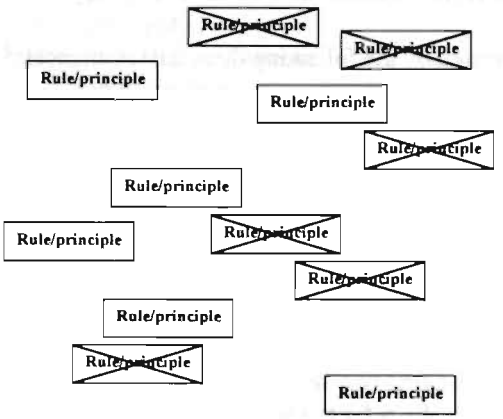

Figure 6: A hybrid rule/principle applies

As a result, the conclusion of the hybrid rule/principle does not follow directly, but only after weighing the reasons arising from the other rules/principles, that are not blocked.

In section 7, this informal sketch of an integrated view on rules and principles will be formalized in Reason-Based Logic. As preliminaries, we discuss how isolated rules/principles, the weighing of reasons, exceptions and conflicts are modeled in Reason-Based Logic (sections 3, 4, 5 and 6, respectively).

3 An isolated rule/principle in Reason-Based Logic

We start our discussion of rules and principles in Reason-Based Logic with the case of an isolated rule/principle. This will be spelled out in detail to illustrate the main elements of Reason-Based Logic.

As an example we use the legal rule that a person driving a car after drinking too much alcohol should be fined a considerable amount of money. (It does not matter that we use an isolated *rule* as an example, since in our view there is no

difference in representation between an isolated rule and an isolated principle.) Assume that we have:

> A person driving a car after drinking too much alcohol should be fined a considerable amount of money.
> John is driving his car after drinking too much alcohol.

If we interpret the first sentence as a rule, the application of this rule must lead to the conclusion:

> John should be fined a considerable amount of money.

This can be represented by the following three RBL sentences:[7]

```
Valid(rule(driving_with_alcohol(person),
    should_be_fined(person)))
Driving_with_alcohol(john)
Should_be_fined(john)
```

For this representation it does not matter whether the RBL rule concerning driving with alcohol stems from a rule or from a principle: both rules and principles are represented in Reason-Based Logic as RBL rules.

We show that if the first two sentences are assumed to be true, the truth of the third sentence follows. We refer to the relations between facts, such as EXCLUSION and WEIGHING, as discussed in chapter 2, section 5. Instead of using the nonmonotonic rules of inference (chapter 2, section 6) and the corresponding technicalities of extensions, we make some 'normality assumptions', such as that a rule is not excluded.

First we note that the condition of the RBL rule concerning driving with alcohol is satisfied because Driving_with_alcohol(john) is assumed to be true.[8]

The first normality assumption is that the rule is not excluded:

```
¬Excluded(rule(driving_with_alcohol(person),
        should_be_fined(person)),
    driving_with_alcohol(john),
    should_be_fined(john))
```

Since there are no facts that lead to the exclusion of the rule, this assumption is reasonable.

---

[7] Other formalizations are possible. The translation from natural to formal language is a problem that we do not discuss here.

[8] Recall the convention on the translation from formulas to terms (chapter 2, section 4.3).

Using this assumption, the rule is applicable because of the relation between facts called EXCLUSION:

```
Applicable(rule(driving_with_alcohol(person),
        should_be_fined(person)),
    driving_with_alcohol(john),
    should_be_fined(john))
```

APPLICABILITY makes that the fact that satisfies the condition of the rule is a reason for the rule's application:

```
Reason(driving_with_alcohol(john),
    applies(rule(driving_with_alcohol(person),
            should_be_fined(person)),
        driving_with_alcohol(john),
        should_be_fined(john)))
```

In order to use WEIGHING to conclude that the rule applies, we have to make another normality assumption, namely that there is no reason against the application of the rule:[9]

```
¬∃fact_against_application:
    Reason(fact_against_application,
        ¬applies(rule(driving_with_alcohol(person),
                should_be_fined(person)),
            driving_with_alcohol(john),
            should_be_fined(john)))
```

By WEIGHING_AXIOMS we have

```
Outweighs({driving_with_alcohol(john)},
    {},
    applies(rule(driving_with_alcohol(person),
            should_be_fined(person)),
        driving_with_alcohol(john),
        should_be_fined(john)))
```

---

[9] The appearance of the following sentence may suggest that the quantification over the variable *fact_against_application* is only over a specific part of the domain, namely only over those terms that correspond to facts. However, the quantification is over the whole domain. By the definition of a language of Reason-Based Logic (chapter 2, section 4), a similar effect is obtained: a language contains no sentences of the form Reason(*fact*, *state_of_affairs*) in which *Fact* does not correspond to an instance of the condition of some rule rule(*condition*, *conclusion*).

and therefore also, by WEIGHING,

Applies(rule(driving_with_alcohol(*person*),
should_be_fined(*person*)),
driving_with_alcohol(john),
should_be_fined(john))

Using APPLICATION, the rule concerning driving with alcohol now gives,

Reason(driving_with_alcohol(john),
should_be_fined(john))

We have to make a third normality assumption, namely that there are no reasons against Should_be_fined(john):

¬∃*fact_against_fining*:
Reason(*fact_against_fining*,
¬should_be_fined(john))

Using WEIGHING_AXIOMS and WEIGHING a second time we find that

Outweighs({driving_with_alcohol(john)},
{},
should_be_fined(john))

and finally that

Should_be_fined(john)

are true.

At three steps in the discussion above, we had to make a normality assumption. In summary, we assumed that

- The rule is not excluded.
- There is no reason against application of the rule.
- There is no reason against fining John.

These assumptions can be avoided using the nonmonotonic inference rules of Reason-Based Logic discussed in chapter 2, section 6. It can be shown that the theory consisting of the sentences

Valid(rule(driving_with_alcohol(*person*),
should_be_fined(*person*)))
Driving_with_alcohol(john)

has a unique extension that contains

Should_be_fined(john)

and does not contain sentences contradicting the assumptions. The extension is the closure under RBL-deduction (definition 7 in chapter 2, section 6) of the set that consists of the following sentences:

```
Valid(rule(driving_with_alcohol(person),
    should_be_fined(person)))
Driving_with_alcohol(john)
Applicable(rule(driving_with_alcohol(person),
        should_be_fined(person)),
    driving_with_alcohol(john),
    should_be_fined(john))
Reason(driving_with_alcohol(john),
    applies(rule(driving_with_alcohol(person),
            should_be_fined(person)),
        driving_with_alcohol(john),
        should_be_fined(john)))
Outweighs({driving_with_alcohol(john)},
    {},
    applies(rule(driving_with_alcohol(person),
            should_be_fined(person)),
        driving_with_alcohol(john),
        should_be_fined(john)))
Applies(rule(driving_with_alcohol(person),
        should_be_fined(person)),
    driving_with_alcohol(john),
    should_be_fined(john))
Reason(driving_with_alcohol(john),
    should_be_fined(john))
Outweighs({driving_with_alcohol(john)},
    {},
    should_be_fined(john))
Should_be_fined(john)
```

In the remainder of this chapter, we do not mention normality assumptions or extensions.

4 Weighing reasons in Reason-Based Logic

In this section we describe an example of weighing reasons in Reason-Based Logic in detail. We assume that the following sentences are true:

Robbing someone should be punished.
John has robbed Peter.

If we interpret the first sentence as a principle, we obtain a reason why John should be punished. Since there are no other reasons, it follows that John should be punished.

Now assume that the following sentences are also true:

Minor first offenders should not be punished.[10]
John is a minor first offender.

We find a second reason relevant concerning punishing John, but this time a reason against the fact that John should be punished.

So, there is a conflict of reasons. Without further information, Reason-Based Logic does not enforce the conclusion that John should be punished or that he should not. Both are possible. Only if it is true that one of the reasons outweighs the other, a conclusion follows.

We assume that the reason that John is a minor first offender outweighs the reason that John has robbed Peter:

'John is a minor first offender' as a reason for not punishing John outweighs the reason 'John has robbed Peter'.

In Reason-Based Logic this can be represented as follows:

Valid(rule(robbed(*person1*, *person2*),
  should_be_punished(*person1*)))[11]
Robbed(john, peter)

---

[10] In the natural language version of this sentence it is ambiguous what the scope of 'not' is. As the formal version shows, we mean 'It should not be the case that minor first offenders are punished', and not 'It should be the case that minor first offenders are not punished'.

[11] A representation of the condition of this rule that is somewhat closer to its natural language counterpart would be ∃*person2*: Robbed(*person1*, *person2*). However for simplicity the definition of the language of RBL (chapter 2, section 4) prohibits quantifiers in the conditions of rules. The condition of the rule without the existential quantifier as it is used here leads to similar consequences as the condition with the quantifier, since it can only be fulfilled if the variable *person2* is instantiated.

Valid(rule(is_minor_first_offender(*person*),
¬should_be_punished(*person*)))
Is_minor_first_offender(john)
Outweighs({is_minor_first_offender(*person1*)},
{robbed(*person1*, *person2*)},
¬should_be_punished(*person1*))

Using similar normality assumptions as in the example of section 3, it can be shown that both rules apply:

Applies(rule(robbed(*person1*, *person2*),
should_be_punished(*person1*)),
robbed(john, peter),
should_be_punished(john))
Applies(rule(is_minor_first_offender(*person*),
¬should_be_punished(*person*)),
is_minor_first_offender(john),
¬should_be_punished(john))

Applying the two rules leads to two reasons, one for and one against punishing John:

Reason(robbed(john, peter),
should_be_punished(john))
Reason(is_minor_first_offender(john),
¬should_be_punished(john))

Assuming that there are no other relevant reasons for punishing John, and using the information about the relative weight of the reasons, the relation between facts WEIGHING gives:

¬Should_be_punished(john)

It can be the case that additional reasons give rise to another conclusion. We will discuss what can happen if there is a second reason for punishing John. We add the following facts:

Injuring someone should be punished.
John has injured Peter.

These can be represented as:

Valid(rule(injury,
injured(*person1*, *person2*),
should_be_punished(*person1*)))
Injured(john, peter)

Now a second reason for punishing John arises:

Reason(injured(john, peter),
should_be_punished(john))

As a result, we cannot make the assumption that there are no other reasons for punishing John than Robbed(john, peter).

Right now, WEIGHING cannot be used to conclude whether John has to be punished or not, since there is no information about how the reasons are to be weighed.

It is possible that the additional reason does not change the result of weighing: the reason against punishing outweighs the two reasons for punishing. It should be noted that to reach a conclusion it does not suffice that the reason against punishing outweighs each of the two for punishing on its own. In that case,

Outweighs({is_minor_first_offender(*person1*)},
{injured(*person1*, *person2*)},
¬should_be_punished(*person1*))

is also true, but WEIGHING can still not be used: that would require weighing information about all three reasons together. In order to use WEIGHING it is required that

Outweighs({is_minor_first_offender(*person1*)},
{robbed(*person1*, *person2*), injured(*person1*, *person2*)},
¬should_be_punished(*person1*))

is true. In that case the conclusion that John should not be punished follows (using an appropriate normality assumption).

An interesting case, characteristic for reasoning with reasons, occurs if the two reasons for punishing John together outweigh the reason against punishing him:

Outweighs({robbed(*person1*, *person2*), injured(*person1*, *person2*)},
{is_minor_first_offender(*person1*)},
should_be_punished(*person1*))

In this case, WEIGHING leads to the opposite conclusion, viz. that John should be punished:

, Should_be_punished(john)

The two pros can together outweigh the con, even if each pro on its own is outweighed by the con. This phenomenon has been called *accrual of reasons*.[12]

5 Exceptions in Reason-Based Logic

In this section, we show how exceptions can be modeled in Reason-Based Logic. We say that there is an *exception* to a rule (or principle), if the rule's (or principle's) condition is satisfied while its conclusion does not hold. It can be the case that there is another rule/principle the conclusion of which is incompatible with the conclusion of the rule/principle under consideration. In that case we speak of a conflict of rules/principles.[13] Conflicts of rules/principles are discussed in the next section.

In Reason-Based Logic, there are two main mechanisms to model exceptions to a rule/principle, namely by exclusionary reasons and by reasons against the application of a rule.[14] We discuss these in the following two subsections.

5.1 Exceptions and exclusionary reasons

Legal rules often, if not always, have scope restrictions that are not explicitly mentioned in the rule itself. For instance, in the legal rule that we already encountered about driving with alcohol,

> A person driving a car after drinking too much alcohol should be fined a considerable amount of money.

it is not explicitly mentioned in which country the rule is valid. It may be objected that this is due to the particular formulation chosen here, but also in the literal wordings in a statute the country will normally not be mentioned at all, or only in a separate section, where it is stated that the articles in the statute are only valid in a particular country.

In Reason-Based Logic, exclusionary reasons can be used to model implicit scope restrictions. For instance,

---

[12] Pollock (1991a, p. 51) uses this term. He writes that it is a natural supposition that reasons accrue, but then surprisingly rejects it. We come back to Pollock's opinion in chapter 6, section 2.

[13] Cf. the distinction between undercutting and rebutting exceptions (Pollock, 1987-1995): in both cases there is an exception to a rule/principle, but in case of a rebutting exception there is also a conflict of rules/principles.

[14] The first mechanism has counterparts in many logical formalisms (cf. Prakken's (1993b, p. 84ff.) overview of exceptions), the second is typical for Reason-Based Logic.

Valid(rule(in_country(*country*) ∧ ¬*country* = holland,
excluded(rule(driving_with_alcohol(*person*),
should_be_fined(*person*)))))[15]

will have the effect that if John was driving in Germany, represented as

In_country(germany),

the rule concerning driving with alcohol is excluded. As a result, the rule concerning driving with alcohol is not applicable, does not apply, and does not lead to the conclusion that John should be fined. (Of course, it is possible that the same conclusion nevertheless follows due to another valid rule, e.g., a German rule of law, that is not excluded.)

Scope restrictions for a class of rules can be represented by explicit knowledge on the origin of the rules. For instance, using the explicit knowledge on which articles rules are based and which articles are in the penal code, all rules that are based on articles in the penal code are restricted to Holland by the following:

Valid(rule(in_country(*country*) ∧ ¬ *country* = holland
∧ based_on(*rule*, *article*) ∧ in_penal_code(*article*),
excluded(*rule*)))

An obvious objection to this type of representation of exceptions, viz. *outside* the rule, is that since they are often explicitly available they can be made part of the rule during the translation of the legal rule to its formal counterpart. For instance, this would lead to the following representation:

Valid(rule(driving_with_alcohol(*person*) ∧ in_country(holland),
should_be_fined(*person*)))

There are drawbacks to this approach, as is generally accepted (see chapter 4). First, it can easily lead to very long rule conditions, most of which have to be repeated in many rules and are almost always unimportant for a particular case. For analogous reasons, in actual codifications of legal rules scope restrictions are not explicitly stated in each rule. Second, the dissimilarity in structure of the informal and the formal representation is unnecessarily enlarged.[16] As a result, translation in either direction becomes harder, which is particularly a problem in a

[15] We have made a simplification here, since facts are often dependent on a situation or case. For instance, a rule can apply to *a case*. As a result, many predicates would need an extra variable for cases. For convenience we leave cases implicit. For instance, in the following In_country(*country*) means that the case *at hand* is in the country represented by *country*.

[16] Cf. the desirability of an *isomorphic* representation of the law (see e.g. Bench-Capon and Coenen, 1992).

constantly changing domain, such as the law. Third - and this is a drawback that cannot be overcome - not all exceptions to legal rules are explicitly available since it is impossible to anticipate all cases in which a rule is not applicable.

The third point brings us to the second way of representing exceptions in Reason-Based Logic.

5.2 Exceptions and reasons against application

When a legal rule is made by the legislator, not all cases that fall inside the definition set by a legal rule can be foreseen, if not fundamentally, then at least in practice. We do not treat the philosophical side of these matters, but give a concrete example.

It can happen that there is a case that falls within the rule's condition and to which the rule is applicable, but to which the rule should not apply for some other reason. For instance application of the rule might be against its purpose.

We assume that there is a rule that forbids sleeping in the railway station. The rule has as its purpose to prevent tramps from occupying the station as their place to spend the night. An old lady that wants to meet a friend at the station dozes off when the evening train turns out to be late. Should the prohibition apply to this lady?[17]

The following two sentences describe the case:

Valid(rule(sleep_in_station(*act*),
    forbidden(*act*)))
Sleep_in_station(lady's_act)

We assume that application of the rule about the sleeping prohibition in the case of the lady is against the rule's purpose:

Application_against_purpose(rule(sleep_in_station(*act*),
        forbidden(*act*)),
    sleep_in_station(lady's_act),
    forbidden(lady's_act))

Hence, we need a general rule stating that if application is against the purpose of a rule, this is a reason not to apply the rule:

Valid(rule(application_against_purpose(*rule*, *fact*, *state_of_affairs*),
    ¬applies(*rule*, *fact*, *state_of_affairs*)))

[17] This example is inspired by Fuller's (1958, p. 664). The formulation here is taken from Hage and Verheij (1994a, b).

Since the condition of the rule about the sleeping prohibition is satisfied, we have a reason to apply it (by APPLICABILITY):

Reason(sleep_in_station(lady's_act),
  applies(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act)))

But we also have a reason against application:

Reason(application_against_purpose(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act)),
  ¬applies(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act)))

We suppose that the reason against application of the rule because of its purpose outweighs the reason for application because of the applicability of the rule:

Outweighs({application_against_purpose(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act))},
  {sleep_in_station(lady's_act)},
  ¬applies(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act))

We now conclude

¬Applies(rule(sleep_in_station(*act*),
    forbidden(*act*)),
  sleep_in_station(lady's_act),
  forbidden(lady's_act))

Because the rule about the sleeping prohibition is not applied, it does not lead to the prohibition of the lady's sleeping.

6 Conflicts in Reason-Based Logic

We speak of a conflict of rules/principles, if there is a group of rules/principles the conclusions of which are incompatible, while their conditions are satisfied. There are two main mechanisms in Reason-Based Logic to deal with conflicts of rules and principles, namely by means of exclusionary reasons and by means of weighing reasons.

6.1 Conflicts and exclusionary reasons

When dealing with conflicting legal rules, several types of so-called *conflict rules* are used in law: specific priority clauses for pairs of rules, or for classes of rules, and general rules such as Lex Superior, Lex Posterior, and Lex Specialis. The effect is that one or more of the conflicting rules are excluded and that in the end there is no conflict left.

Such conflict rules can be represented in Reason-Based Logic by means of exclusionary reasons. For instance, following Prakken (1993b), if there is a contract with features of lease of business accommodation and of another type of contract, and there is a conflict between a legal rule dealing with such lease contracts and one dealing with contracts of the other type, the first rule prevails according to Section 7A: 1624 of the Dutch civil code. This legal rule might be represented as follows:

Valid(rule(deals_with_lease_of business_accommodation(*rule1*)
∧ applies(*rule1*)
∧ deals_with_contracts_of_another_type(*rule2*)
∧ in_conflict(*rule1*, *rule2*),
excluded(*rule2*)))

More generally, explicit knowledge about prevalence can be used, for instance:

Valid(rule(applies(*rule1*)
∧ in_conflict(*rule1*, *rule2*)
∧ prevails_over(*rule1*, *rule2*),
excluded(*rule2*)))

Using the latter rule about prevalence, a conflict rule such as Lex Posterior can be represented as follows:

Valid(rule(more_recent(*rule1*, *rule2*),
prevails_over(*rule1*, *rule2*)))

It has to be specified when rules are in conflict. It can for instance be specified that rules are in conflict when they have opposite conclusions.[18]

In practice, it can happen that conflict rules are themselves involved in a conflict. For instance, a rule can be of earlier date and of higher authority than another rule. Since the conflict rules are themselves represented as rules in Reason-Based Logic, such conflicts of conflict rules can be approached in the same way as conflicts in general.

6.2 Conflicts and weighing reasons

The second mechanism to deal with conflicting rules/principles is by the weighing of the resulting reasons.[19] An example was already discussed in section 4 of this chapter.

Not all rules and principles involved in a conflict lead to conflicting reasons, since there can be rules/principles that do not apply because of exclusionary reasons, or reasons against their application. If after such simplifications of the conflict there is still a conflict of reasons, information about their relative weight can resolve the conflict and lead to a final conclusion. So, there are several layers in which a conflict of rules/principles is simplified before the resulting reasons are weighed. Figure 7 gives an overview.

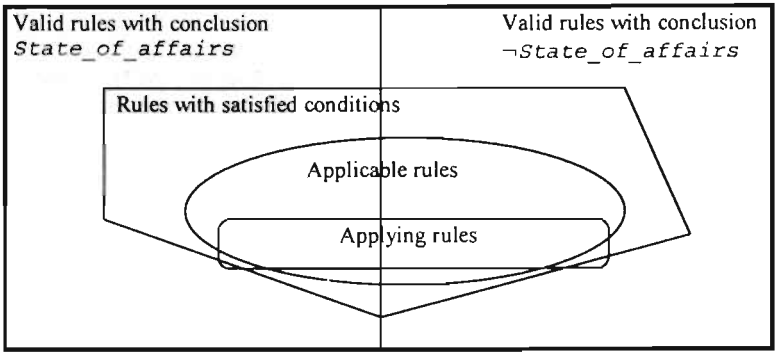

Figure 7: Not all conflicting RBL rules lead to conflicting reasons.

It may seem strange that the applying RBL rules are not indicated as a subset of the applicable rules. In section 8.3 on the analogous application of a rule, we will see an example of an RBL rule that applies, while it is not applicable.

---

[18] The need for specifying when rules are in conflict can be considered a drawback since it puts a heavy burden on the domain theory. However, it can also be considered an advantage since it can make the notion of conflict more manageable.

[19] This mechanism can only deal with conflicts of rules/principles with opposite conclusions, due to the notion of weighing as modeled in Reason-Based Logic.

As a final remark about dealing with conflicts of rules/principles in Reason-Based Logic, we stress that Reason-Based Logic does not resolve all conflicts, and merely provides different means to represent conflict-resolving information. For instance, the following set of sentences does not have an extension in Reason-Based Logic due to an unresolved conflict of rules:

A
B
Valid(rule(a, c))
Valid(rule(b, d))
¬C ∨ ¬D

However, there is no inconsistency (in the sense of RBL-deduction), and the conflict is resolved if the sentence Excluded(rule(a, c), a, c) or the sentence Excluded(rule(b, d), b, d) is added.

7 Rules and principles in Reason-Based Logic

We now return to our integrated view on rules and principles, as introduced in section 2. Recall that our view was based on two assumptions:

- Both rules and principles give rise to reasons if they are applied.
- The differences between reasoning with rules and principles result from different types of relationships with other rules and principles, which may interfere.

In section 7.1, we discuss our basic example of the role of the relationships between rules and principles, namely a rule with underlying principles. In section 7.2, we return to the differences between rules and principles as discussed in section 1.

7.1 A rule and its underlying principles

In section 2.1, we discussed the Dutch legal rule of Art. 7A:1612 BW that sale of a house should not terminate an existing rent contract. This rule can be represented in Reason-Based Logic as follows:

Valid(rule(sale_house,
  ought_to_be_done(continuation_contract)))

We considered two principles underlying this rule, namely a pro-principle that somebody who lives in a house should be protected against measures that threaten

the enjoyment of the house, and a con-principle that contracts only bind the contracting parties. These principles can be represented as RBL rules as follows:

```
Valid(rule(protects_inhabitants(act),
    ought_to_be_done(act)))
Valid(rule(¬party_bound_by_contract,
    ¬ought_to_be_done(continuation_contract)))
```

The fact that these principles underlie the rule of Art. 7A:1612 BW is represented as:

```
Underlies(rule(protects_inhabitants(act),
        ought_to_be_done(act)),
    rule(sale_house,
        ought_to_be_done(continuation_contract)))
Underlies(rule(¬party_bound_by_contract,
        ¬ought_to_be_done(continuation_contract)),
    rule(sale_house,
        ought_to_be_done(continuation_contract)))
```

The rule and its underlying principles are schematically shown in Figure 8.

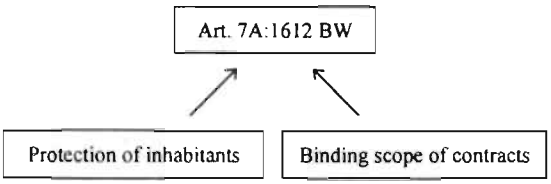

Figure 8: The rule of Art. 7A:1612 BW and its underlying principles

If a house with renting inhabitants is sold, the two principles lead to conflicting reasons, since continuation of an existing rent contract protects the inhabitants of a house, while the new owner is not bound by the contract with the inhabitants. We have

```
Protects_inhabitants(continuation_contract)
¬Party_bound_by_contract
```

and therefore the two RBL rules about the protection of inhabitants and about the binding scope of contracts lead to the conflicting reasons:

```
Reason(protects_inhabitants(continuation_contract),
    ought_to_be_done(continuation_contract))
```

Reason(¬party_bound_by_contract,
¬ought_to_be_done(continuation_contract))

However, by making the legal rule of Art. 7A:1612 BW, the legislator has balanced the conflicting principles, and decided how the reasons generated by them should be weighed against each other. Therefore, if we have the fact

Sale_house

the rule of Art. 7A:1612 BW should lead to the conclusion

Ought_to_be_done(continuation_contract)

without the interference of the two underlying principles: the rule of Art. 7A:1612 BW replaces its underlying principles if it applies (see section 2.1), and the two principles should not be applicable. The required situation is shown in Figure 9.

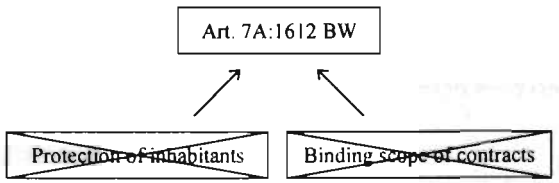

Figure 9: The rule of Art. 7A:1612 BW replaces its underlying principles if it applies

In Reason-Based Logic, replacement can be modeled using exclusionary reasons. We need the following rule:

Valid(rule(underlies(*rule1*, *rule2*) ∧ applies(*rule2*),
excluded(*rule1*)))[20]

Since we can conclude

Applies(rule(sale_house,
ought_to_be_done(continuation_contract)),
sale_house,
ought_to_be_done(continuation_contract))

[20] Henry Prakken has correctly noted that *rule2* also excludes *rule1* in case there is another rule or principle that does not underlie *rule2*, but nevertheless interferes. As a result, there can be no interaction of the other rule or principle with *rule1* if *rule2* applies. This does not always seem desirable, and deserves further study. Interestingly, in this situation *rule2* is not a typical rule.

we find:

Excluded(rule(protects_inhabitants(*act*),
ought_to_be_done(*act*)),
protects_inhabitants(continuation_contract),
ought_to_be_done(continuation_contract))
Excluded(rule(¬party_bound_by_contract,
¬ought_to_be_done(continuation_contract)),
¬party_bound_by_contract,
¬ought_to_be_done(continuation_contract))

The principles about the protection of inhabitants and about the binding scope of contracts do no longer lead to reasons. As a result, the rule of Art. 7A:1612 BW leads without interference to the conclusion

Ought_to_be_done(continuation_contract),

just as required.

7.2 The differences between rules and principles

We can now finish our integrated view on rules and principles as represented in Reason-Based Logic. As in the case of a rule that replaces its underlying principle, a typical rule is an RBL rule that leads to exclusionary reasons against the applicability of any interfering rule or principle. A typical principle is an RBL rule that does not exclude any interfering rule/principle. Interfering rules and principles are typically rules and principles with equal or opposite conclusion.

This is in line with our two main assumptions:

- Both rules and principles give rise to reasons if they are applied. The difference between the two is that an applying rule not only generates a reason for its conclusion, but also exclusionary reasons for the principles it replaces.
- The differences between reasoning with rules and reasoning with principles result from different types of relationships with other rules and principles, interfering with them: rules lead to exclusionary reasons to interfering rules and principles, while principles lead to reasons that are weighed in case of a conflict.

It is clear that in this view there is no clear border between rules and principles. For instance, an isolated rule cannot be distinguished from an isolated principle. Only if there are interfering rules and principles, gradual differences can be seen. On the one extreme there is the typical principle that, if it applies, does not generate exclusionary reasons for any of the rules and principles that interfere with it. On the other extreme there is the typical rule that, if it applies, excludes all

interfering rules and principles. In between the two extremes there are many degrees of hybrid rules/principles, some more principle-like, others more rule-like.

In section 1, we discussed three differences between rules and principles. First, it seemed that rules lead directly to their conclusion if they apply, while principles lead to reasons that have to be weighed. This difference has disappeared since in our view both rules and principles generate reasons. Therefore both rules and principles first lead to reasons that are then weighed. Nevertheless, also in our view, rules *seem* to lead directly to their conclusion. This is the result of the fact that in the case of an applying rule no weighing of reasons is necessary since all interfering rules and principles are excluded. Therefore, the step from reason to conclusion seems immediate.

Second, it seemed that conflicting rules lead to a contradiction if they apply, while conflicting principles merely lead to conflicting reasons. In our representation, no real contradiction can arise by the application of rules with opposite conclusions, since rules just as principles only generate reasons. Moreover if an apparent rule gives rise to a reason that conflicts with another reason, this is a sign that it is *not* a typical rule, but has a somewhat more principle-like character.

Third, it seemed that rules lead to their conclusion in isolation, while principles interact with other principles: additional relevant reasons arising from other principles can influence the result of weighing. In our view, this seeming difference is beside the point since rules in isolation do not differ from principles in isolation. The rule-like character of a rule can only be appreciated if there are interfering rules or principles.

8 Analogy in Reason-Based Logic

The last topic that we discuss is reasoning by analogy.[21] As an application of our integrated view on rules and principles, we describe three different ways of reconstructing reasoning by analogy. To avoid misunderstanding, we stress that our approach to reasoning by analogy is not based on cases,[22] but on rules and principles. Instead of using the similarity and dissimilarity of cases as criteria to justify reasoning by analogy, we use the relationships between rules and principles.

We assume that in reasoning by analogy there is a rule that does not apply because its condition is not satisfied, but that nevertheless its conclusion holds on the basis of additional information about the relationships between the rule and other rules and principles. We distinguish three forms of reasoning to analyze reasoning by analogy:

---

21 This section is based on Verheij and Hage (1994).

22 See, for instance, Ashley (1990), Yoshino *et al.* (1993) and Tiscornia (1994).

- Application of principles that underlie the original rule that does not apply itself.
- Application of an analogous rule/principle that has the same underlying principles as the original rule that does not apply.
- Analogous application of the original rule, i.e., application of the rule with a 'non-standard' justification, based on, for instance, a principle.

We do not claim that these three forms of reasoning are always cases of reasoning by analogy, but that they are useful means to analyze a given case of reasoning by analogy. Below we use one example, and analyze it by the three mentioned forms of reasoning.

8.1 Application of underlying principles

In the first form of reasoning by analogy, the principles apply that underlie the original rule that does not apply itself.

The example we use is based on Art. 7A:1612 BW. It was also used in the sections 2.1 and 7.1 to explain the replacement of the principles underlying a rule. Again, we have one rule and two underlying principles:

```
Valid(rule(sale_house,
    ought_to_be_done(continuation_contract)))
Valid(rule(protects_inhabitants(act),
    ought_to_be_done(act)))
Valid(rule(¬party_bound_by_contract,
    ¬ought_to_be_done(continuation_contract)))
Underlies(rule(protects_inhabitants(act),
        ought_to_be_done(act)),
    rule(sale_house,
        ought_to_be_done(continuation_contract)))
Underlies(rule(¬party_bound_by_contract,
        ¬ought_to_be_done(continuation_contract)),
    rule(sale_house,
        ought_to_be_done(continuation_contract)))
```

Here we assume that a house with renting inhabitants is not sold, but donated. So, we have the facts:

```
¬Sale_house
Donation_house
```

As a result, the condition of the rule of Art. 7A:1612 BW is not satisfied, and the rule does not apply. But just as in the case of sale, continuation of the existing rent

contract is a way to protect the inhabitants, while the new owner is not bound by the existing contract:

Protects_inhabitants(continuation_contract)
¬Party_bound_by_contract

Therefore, the conditions of the principles about the protection of inhabitants and about the binding scope of contracts are satisfied. Since the rule of Art. 7A:1612 BW does not apply, the replacement rule

Valid(rule(underlies(*rule1*, *rule2*) ∧ applies(*rule2*),
excluded(*rule1*)))

does not give exclusionary reasons for the two underlying principles. They apply and give rise to the reasons:

Reason(protects_inhabitants(continuation_contract),
ought_to_be_done(continuation_contract))
Reason(¬party_bound_by_contract,
¬ought_to_be_done(continuation_contract))

The situation is shown in Figure 10.

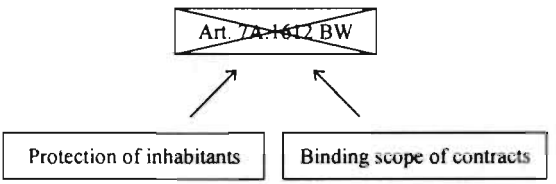

Figure 10: The principles underlying the rule of Art. 7A:1612 BW apply

So, in the case of donation two reasons arise that are based on the same principles as those taken into account by the legislator, when the original rule was made.

There are good reasons to assume that the weighing of these reasons has the same outcome as in the reasoning of the legislator:

Outweighs({protects_inhabitants(continuation_contract)},
{¬party_bound_by_contract},
ought_to_be_done(continuation_contract))

and leads to the same conclusion that the contract should be continued:

Ought_to_be_done(continuation_contract)

In this analysis, two principles applied in the case of donation. They are precisely the two principles that were replaced in the case of sale. The case of donation is therefore in a sense of *the same kind* as the case of sale. Therefore we speak of a form of reasoning by analogy. If only some of the underlying principles apply, or more goals and principles are relevant, we cannot always speak of a case of reasoning by analogy. The case might even be solved differently, since the reasons might be weighed differently.

8.2 Application of an analogous rule/principle

In the second form of reasoning by analogy, a analogous rule/principle applies that has the same underlying principles as the original rule. This leads to another analysis of the same example.

In our example the analogous rule/principle might be:

```
Valid(rule(donation_house,
    ought_to_be_done(continuation_contract)))
```

The legal decision maker that wants to base his reasoning on this rule has to justify its validity. This justification can be based on the same reasons as the ones used by the legislator when he made Art. 7A:1612 BW:

```
Reason(protects_inhabitants(continuation_contract),
    valid(rule(donation_house,
        ought_to_be_done(continuation_contract))))
Reason(¬party_bound_by_contract,
    ¬valid(rule(donation_house,
        ought_to_be_done(continuation_contract))))
```

In this line of reasoning, the two reasons are not relevant for the conclusion that the contract should be continued, but for the validity of the new RBL rule about donation. In their new role, the reasons might be weighed the same way as before:

```
Outweighs({protects_inhabitants(continuation_contract)},
    {¬party_bound_by_contract},
    valid(rule(donation_house,
        ought_to_be_done(continuation_contract)))
```

The conclusion is that the RBL rule about donation is valid.

It may seem that there is a problem here with the separation of powers: while the legislator can make rules, the legal decision maker cannot. However, this problem is only seeming, and due to the different meanings of rule validity in law

and in reasoning. We use the term 'rule validity' in the latter sense. For rule validity in that sense the separation of powers is irrelevant.[23]

If the rule about donation applies, the principles about the protection of inhabitants and about the binding scope of contracts are again replaced by the rule about donation and do not apply. An overview of the relations of the rules and principles involved in this reasoning is shown in Figure 11.

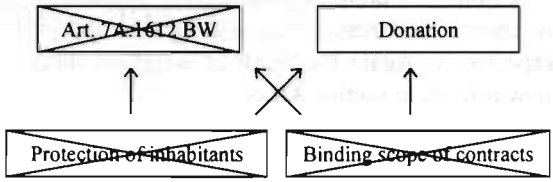

Figure 11: The rule about donation applies having the same underlying principles as the original rule of Art. 7A:1612 BW

Since the rule about donation has the same underlying principles as the rule of Art. 7A:1612 BW we say that a rule is applied analogous to the original rule.

8.3 Analogous application of the original rule

The third form of reasoning by analogy is typical for Reason-Based Logic, since it involves reasons for and against applying a rule.

In this third analysis of the example, the rule of Art. 7A:1612 BW is not applicable, since its condition is not satisfied, just as in the previous two analyses. As a result, the standard reason for applying the rule, based on the relation between facts APPLICABILITY (chapter 2, section 5), does not arise. However, a rule that is not applicable can apply, since there can be other reasons that lead to its application.

In our case, the reasons are again those for and against the continuation of the contract having a new role. They now are represented as follows:

```
Reason(protects_inhabitants(continuation_contract),
    applies(rule(sale_house,
            ought_to_be_done(continuation_contract)),
        sale_house,
        ought_to_be_done(continuation_contract)))
```

[23] In Verheij and Hage (1994), we put it differently: we wrote that the legal decision maker can only validate legal principles (and not legal rules) because of the separation of powers. However, in the line of reasoning described in the text the two underlying principles are replaced if the RBL rule about donation applies. Otherwise the reasons arising from these principles would be taken into account twice. As a result, the RBL rule about donation has a rule-like character.

```
Reason(¬party_bound_by_contract,
    ¬applies(rule(sale_house,
            ought_to_be_done(continuation_contract)),
        sale_house,
        ought_to_be_done(continuation_contract)))
```

Here the reasons protects_inhabitants(continuation_contract) and ¬party_bound_by_contract are reasons for and against applying the rule of Art. 7A:1612 BW, respectively. Again the result of weighing these reasons might be the same in this new role, as in section 8.2:

```
Outweighs({protects_inhabitants(continuation_contract)},
    {¬party_bound_by_contract},
    applies(rule(sale_house,
            ought_to_be_done(continuation_contract)),
        sale_house,
        ought_to_be_done(continuation_contract)))
```

As a result, we can conclude that the rule of Art. 7A:1612 BW applies, even though its condition is not satisfied and it is not applicable:

```
Applies(rule(sale_house,
            ought_to_be_done(continuation_contract)),
    sale_house,
    ought_to_be_done(continuation_contract))
```

Since the rule of Art. 7A:1612 BW applies, it replaces its underlying principles by the replacement rule, just as any applying rule: the principles about the protection of inhabitants and about the binding scope of contracts are excluded and do not apply. Figure 9 shows the relations of the rules and principles involved (but does not show the reasons in their new role). These relations are the same as in the case of normal rule application. Since in this example the rule does apply, but not for the standard reason that its condition is satisfied, we call this *analogous* rule application.

Chapter 4

Formalizing rules: a comparative survey

In the chapters 2 and 3, we have described our approach to formalizing rules: Reason-Based Logic. In this chapter, we discuss a number of other approaches, and compare them to ours. We focus on issues concerning rules that arise because of the defeasibility of arguments.[1]

In section 1, we make some general remarks on rules and their role in argumentation. In section 2, we treat the classic formalization of rules as material conditionals, and to what extent this formalization can cope with a number of issues related to the defeasibility of arguments. Section 3 continues with a discussion of approaches to dealing with the relevance of rule conditions for rule conclusions. We discuss approaches to dealing with exceptions to rules in section 4, and approaches to dealing with rule conflicts in section 5. In section 6, we look at reasoning about rules.

We wish to stress that many of the observations in this chapter are not original.[2] However, we have added some originality by focusing on different issues instead of on specific formalisms. We have selected a number of well-known and influential formalisms, and use them to explain general approaches to the issues. In this way, the approach to formalizing rules of Reason-Based Logic is put in perspective.

1 Rules in argumentation

In this section, we explain our view on rules. We start with the relation between rules and arguments. Some remarks on syllogistic and enthymematic arguments follow. The section ends with a discussion of ordinary rule application.

---

1 Nute (1980) and Sanford (1989) describe other interesting topics, such as counterfactual conditionals.

2 We have especially benefited from the discussions by Haack (1978), Prakken (1993a, chapters 5 and 7) and Makinson (1994).

1.1 Rules and arguments

We recall our interpretation of rules and their relation to arguments (see also chapter 1, section 4.1 and chapter 2, section 1.1). As our starting point, we take informal arguments as they occur in practice, e.g.,

Mary is born in Maastricht.
So, Mary pronounces the letter g softly.
So, people can tell that Mary is from the south of the Netherlands.

We present arguments in an idealized form, with clearly distinguished steps. Each step consists of a reason and a conclusion, as follows:

Reason.
So, *Conclusion.*

Arguments can consist of several steps. In that case, the conclusion of one step is the reason of the next. The example argument consists of two steps. The first step has the reason 'Mary is born in Maastricht' and the conclusion 'Mary pronounces the letter g softly', the second step the reason 'Mary pronounces the letter g softly' and the conclusion 'People can tell that Mary is from the south of the Netherlands'.

The steps in the argument can also occur in other arguments. For instance, the first step in the argument above also occurs in the following argument:

Mary is born in Maastricht.
So, Mary pronounces the letter g softly.
So, people from Amsterdam may find Mary's accent amusing.

In other words, steps in an argument are independent of the particular argument in which they occur. Each step can be used in an argument because there exists some relation between the reason and the conclusion of the step. This relation between reason and conclusion as expressed by the argument step, is what we call a *rule*.

Often argument steps follow a pattern. For instance, the first argument above can be made for anyone who is born in Maastricht. We have the following argument scheme:

Person is born in Maastricht.
So, *Person* pronounces the letter g softly.
So, people can tell that *Person* is from the south of the Netherlands.

The steps in the argument scheme can be used in an actual argument independently of the particular person mentioned. *Person* is a variable, that can be filled in at will: whoever the person *Person* is, Mary, Peter, or Fred, the scheme gives rise to an acceptable argument. Also the relation between reason and conclusion in a step

in such an argument scheme is called a rule, but this time it is a rule with a variable.

There are few things about rules of reasoning that are generally agreed upon. However, a common starting point is that a rule has a condition and a conclusion. The condition and the conclusion of a rule correspond to the reason and the conclusion in an argument step, respectively. So, an argument step of the form

Reason.
So, *Conclusion.*

corresponds to a rule with condition *Reason* and conclusion *Conclusion*. It may seem inconsistent terminology to use two terms, 'reason' and 'condition' for corresponding things. However, there is a difference: if the condition of a rule is used as a reason in an argument, the reason is assumed to hold, while for the validity of a rule it is irrelevant whether its condition holds.

1.2 Syllogistic and enthymematic arguments

If in introductory texts on classical deductive logic examples of informal arguments are given, they typically look as follows (e.g., Purtill, 1979; Copi, 1982, especially p. 235 ff.):

1. John is a thief. If John is a thief, then he should be punished.
   So, John should be punished.

2. Either John is married to Mary or John is married to Edith. John is married to Mary.
   So, John is not married to Edith.

They are used to introduce logical connectives, such as 'If ... then ...' and 'Either ... or ...'. In ordinary language, one also finds the following, closely related arguments that do not contain these connectives:

1'. John is a thief.
   So, John should be punished.

2'. John is married to Mary.
   So, John is not married to Edith.

These arguments result from the arguments 1 and 2 above by omitting one of the premises. From the point of view of classical logic, the first two arguments are complete, while in the second two one of the premises is missing. The arguments 1' and 2' are called *enthymematic*, in contrast with their *syllogistic*

counterparts 1 and 2, that explicitly contain all premises (Copi, 1982, pp. 235, 253).[3]

In this thesis, we have given examples of arguments that resemble the syllogistic type of argument and of arguments that resemble the enthymematic type. This may seem inconsistent. However, the apparent inconsistency disappears if it is noted that the distinction between syllogistic and enthymematic arguments only has meaning *relative to a set of rules*. For instance, the syllogistic arguments above are complete, relative to the rules (or rule schemes) Modus Ponens and Disjunctive Syllogism underlying the argument schemes:

State_of_affairs$_1$. If *State_of_affairs*$_1$, then *State_of_affairs*$_2$.
So, *State_of_affairs*$_2$.

Either *State_of_affairs*$_1$ or *State_of_affairs*$_2$. *State_of_affairs*$_1$.
So, not *State_of_affairs*$_2$.

Relative to these rules, we can distinguish the syllogistic arguments 1 and 2, in which all premises are explicitly stated, and the enthymematic arguments 1' and 2', in which one or more premises are missing.

The example arguments 1' and 2', that are enthymematic with respect to Modus Ponens and Disjunctive Syllogism, are syllogistic with respect to the rules that underlie the argument schemes

Person is a thief.
So, *Person* should be punished.

and

Person$_1$ is married to *Person*$_2$.
So, *Person*$_1$ is not married to *Person*$_3$.[4]

Clearly, our interpretation of rules is closely related to the warrants in Toulmin's (1958) argument scheme.[5]

We have taken some effort to state our interpretation of the notion 'rule' as clearly as possible, for two reasons. First, we think that research on the formalization of reasoning with defeasible arguments should be thoroughly

---

3 The distinction between syllogistic and enthymematic arguments was already made by Aristotle (cf. Copi, 1982).

4 It is sometimes objected that the rules underlying these arguments refer to the meaning of the phrases used. This ignores the fact that also a rule such as Modus Ponens refers to the meaning of its phrases, namely the meaning of 'If ..., then ...', which as we will see is not uncontroversial.

5 Toulmin's argument scheme has recently inspired several researchers (cf., e.g., Bench-Capon, 1995).

grounded in intuitions, simply because that research is inspired by the intuitive differences between actual reasoning and, for instance, deductive reasoning. This is in line with our general method of research (chapter 1, section 7)

Second, different intuitions can cause much confusion. Therefore, we stress that our interpretation of rules differs from several other interpretations in the literature, such as rules of inference, material conditionals, or default rules. Indeed, there is no single, generally accepted interpretation of the notion 'rule'. In fact, a significant part of the research on defeasible reasoning can be regarded as a search for the meaning, or, better, for different meanings of the notion 'rule'.

1.3 Ordinary rule application

In any interpretation of rules, they can in some sense be applied: if there is a rule, the condition of which holds, the conclusion of the rule follows. Here 'holds' and 'follows' can be interpreted in many ways, for instance as 'be true', 'be derivable', or 'be justified by an argument'. The latter interpretation will be our intuitive guideline in this chapter.

Since we will be dealing with several different formalisms, a notational convention is useful. If the conclusion *Conclusion* follows from the assumptions $Assumption_1$, $Assumption_2$, ..., $Assumption_n$, we write:

$$Assumption_1, Assumption_2, ..., Assumption_n \vdash Conclusion$$

Our guiding interpretation of this notation is as follows: assuming $Assumption_1$, $Assumption_2$, ..., $Assumption_n$, the conclusion *Conclusion* is justified (by some argument).

Using this notation, ordinary rule application is denoted as follows:

$$Rule, Condition \vdash Conclusion$$

Here *Rule* denotes that there is a valid rule that has *Condition* as its condition and *Conclusion* as its conclusion.

In First-Order Predicate Logic (see, e.g., Van Dalen (1983) or Davis (1993)), there is an obvious candidate to formalize rules, namely the material conditional.[6] A rule with condition *Condition* and conclusion *Conclusion* can be represented as the material conditional $Condition \rightarrow Conclusion$, and ordinary rule application can be interpreted in two well-known (and equivalent) ways, namely semantically and proof-theoretically:

If *Condition* $\rightarrow$ *Conclusion* and *Condition* are true, then *Conclusion* is true.
From *Condition* $\rightarrow$ *Conclusion* and *Condition*, *Conclusion* is derivable.

---

6 The material *conditional* is often called the material *implication*. Sanford (1989), joining Quine, explains why this is uncareful use of language.

These are usually formally represented as follows:

Condition → *Conclusion*, *Condition* ⊨ *Conclusion*
Condition → *Conclusion*, *Condition* ⊢ *Conclusion*

In our notational convention, both become:

Condition → *Conclusion*, *Condition* ⊢ *Conclusion*

We stress that the symbol ⊢ does not give preference to a semantically or a syntactically defined consequence relation.

In the chapters 2 and 3, we discussed another candidate to formalize rules, namely the rule of Reason-Based Logic. In comparison with the complexity of the rule of Reason-Based Logic, the material conditional is attractively simple. Therefore an important question arises. Why is the material conditional approach to rules unsatisfactory? That is the subject of the next section.

2 Rules as material conditionals

In this section, we discuss the material conditional approach to rules. First we discuss the relevance of rule conditions for rule conclusions and the paradoxes of the material conditional. Then we discuss the behavior of the material conditional with respect to exceptions and conflicts. The section ends with a discussion of the problems of the material conditional related to reasoning about rules.

2.1 Relevance and the paradoxes of the material conditional

If we formalize rules as material conditionals, the first problems that we encounter concern the relevance of the condition for the conclusion.

The rule of our example above, that allowed the argument steps of the scheme

Person is born in Maastricht.
So, *Person* pronounces the letter g softly.

shows the relevance of the condition of a rule for its conclusion. The fact that someone is born in Maastricht is *relevant* for the fact that someone pronounces the letter g softly, in the sense that under normal circumstances the second follows *because* the first holds. This relevance is a consequence of the way the world is: people born in Maastricht, normally pronounce the letter g softly. As a result, the demand of the relevance of a rule's condition for its conclusion is in principle a matter of the domain theory.

For instance, a domain theory that contains a rule with condition 'The sky is blue' and conclusion 'Amsterdam is the capital of the Netherlands' does not meet

the relevance demand. However, the relevance demand is not only a matter of the domain theory, but also of the allowed inferences. We show this using the material conditional as an example. It turns out that material conditionals have properties that are not in line with the relevance demand.

For instance, if we assume that Mary is *not* born in Maastricht, the material conditional with condition Mary_is_born_in_Maastricht and conclusion Mary_pronounces_the_letter_g_softly follows:

¬Mary_is_born_in_Maastricht ⊢ Mary_is_born_in_Maastricht →
Mary_pronounces_the_letter_g_softly

In fact, any material conditional with condition Mary_is_born_in_Maastricht follows, for instance:

¬Mary_is_born_in_Maastricht ⊢ Mary_is_born_in_Maastricht →
¬Mary_pronounces_the_letter_g_softly
¬Mary_is_born_in_Maastricht ⊢ Mary_is_born_in_Maastricht →
There_is_life_on_Mars
¬Mary_is_born_in_Maastricht ⊢ Mary_is_born_in_Maastricht →
¬Mary_is_born_in_Maastricht

The examples have been chosen in such a way that the conditions of the material conditionals become decreasingly relevant for their conclusions. Interpreted as rules that give rise to acceptable arguments, these material conditionals become increasingly absurd. For instance, in our interpretation, the last example reads as follows. Assuming that Mary is not born in Maastricht, there is a rule that makes the argument

Mary is born in Maastricht.
So, Mary is not born in Maastricht.

acceptable.

These examples are due to the first of the following so-called paradoxes of the material conditional (cf., e.g., Haack, 1978, p. 37):

$\neg A \vdash A \rightarrow B$
$B \vdash A \rightarrow B$
$\vdash (A \rightarrow B) \vee (B \rightarrow A)$

Examples of the second are:

Mary_pronounces_the_letter_g_softly ⊢ Mary_is_born_in_Amsterdam →
Mary_pronounces_the_letter_g_softly

$$\text{Mary\_pronounces\_the\_letter\_g\_softly} \vdash \text{There\_is\_life\_on\_Mars} \rightarrow \text{Mary\_pronounces\_the\_letter\_g\_softly}$$

Interpreting the latter, we find: assuming that Mary pronounces the letter g softly, there is a rule that makes the argument

There is life on Mars.
So, Mary pronounces the letter g softly.

acceptable.

An example of the third paradox is:

$$\vdash (\text{There\_is\_life\_on\_Mars} \rightarrow \text{Mary\_pronounces\_the\_letter\_g\_softly}) \vee (\text{Mary\_pronounces\_the\_letter\_g\_softly} \rightarrow \text{There\_is\_life\_on\_Mars})$$

Interpreting this, we find that there is either a rule that makes the argument

There is life on Mars.
So, Mary pronounces the letter g softly.

acceptable, or a rule that makes the argument

Mary pronounces the letter g softly.
So, there is life on Mars.

acceptable.

The examples show that the material conditional does not behave well with regard to relevance. Even if we are careful and assume only material conditionals which have conditions that are relevant for their conclusions, we obtain many other material conditionals for free which lack that property. This has been recognized for long, and is generally considered a drawback of the formalization of rules as material conditionals. For instance, the paradoxes of the material conditional led C.I. Lewis to the definition of the strict conditional (that turned out to have similar paradoxes of its own),[7] and Anderson and Belnap to the development of their logic of relevance.[8]

Some approaches to dealing with relevance are discussed in section 3.

7 Cf. Haack (1978, p. 37) and Sanford (1989, p. 68ff.).
8 Cf. Haack (1978, p. 37, p. 198ff.) and Sanford (1989, p. 129ff.).

2.2 Exceptions to rules

Another source of problems for the material conditional are exceptions to rules. We have already seen several examples of exceptions in the previous chapters (chapter 1, section 4.1, chapter 2, section 1.2, chapter 3, section 5).

There are two intuitive requirements for reasoning with rules with exceptions:

STANDARD CASE
If there is a rule the condition of which holds, then the rule's conclusion follows.

EXCEPTION CASE
If there is a rule the condition of which holds, and there is an exception to the rule, then the rule's conclusion does not follow.[9]

If we model rules as material conditionals, we get the following:

STANDARD CASE
Condition, *Condition* $\rightarrow$ *Conclusion* $\vdash$ *Conclusion*

EXCEPTION CASE
Condition, *Condition* $\rightarrow$ *Conclusion*, *Exception* $\nvdash$ *Conclusion*

The latter is clearly false. We recall the property called monotonicity:

If *Assumptions* $\vdash$ *Conclusion*,
then *Assumptions*, *More_assumptions* $\vdash$ *Conclusion*.

It follows immediately that a reasoning formalism that meets the two requirements above cannot be monotonic. Since First-Order Predicate Logic is monotonic, we conclude that reasoning with rules with exceptions cannot be represented in it.

It may at first seem strange, but the requirement in the standard case, makes reasoning with rules with exceptions nonmonotonic, and not the requirement in the exception case. In the standard case, one jumps to the conclusion of the rule, while there might be an exception. It would be more careful to add the assumption that there is no exception, as follows:

CAREFUL STANDARD CASE
If there is a rule the condition of which holds, and there is no exception, then the rule's conclusion follows.

Clearly, this careful requirement does not lead to nonmonotonicity. For instance, the deductive consequence relation of Reason-Based Logic (chapter 2, beginning

---

9 Of course, the rule's conclusion can hold, as a result of *other* information.

of section 6) is careful in this sense. However, as we already discussed there, this carefulness leads to a weak notion of consequence.

Other approaches to dealing with exceptions to rules are discussed in section 4.

2.3 Rule conflicts

A third source of problems for the formalization of rules as material conditionals are rule conflicts. We have already seen several examples in the previous chapters (chapter 1, section 4.2, chapter 2, section 1.3, chapter 3, section 6). We mention two types of unwanted behavior of the material conditional.

The first type of unwanted behavior is that, if there is a conflict of material conditionals, i.e., their conclusions are incompatible and their conditions satisfied, anything follows. Formally,

$$Condition_1, Condition_2, Condition_1 \rightarrow Conclusion, Condition_2 \rightarrow \neg Conclusion \vdash Anything$$

For instance, interpreting rules as material conditionals, we find: if thieves are punishable, minor first offenders are not punishable, and John is a minor thief, then Fermat's theorem is true. This easy way of settling Fermat's theorem is of course useless since we can also conclude that it is false. Clearly, this behavior of the material conditional is unwanted if one accepts the existence of rule conflicts. Intuitively, a conflict of rules should not lead to a contradiction from which anything follows. We have the following intuitive property:

RULE CONFLICT

If there are rules with incompatible conclusions, the conditions of which hold, no contradiction follows.

The second type of unwanted behavior of the material conditional occurs even if the conditions of rules with incompatible conclusions are not satisfied. We have the following:

$$Condition_1 \rightarrow Conclusion, Condition_2 \rightarrow \neg Conclusion \vdash Condition_2 \rightarrow \neg Condition_1$$

For instance, if thieves are punishable and minor first offenders are not, then minor first offenders are not thieves. It would be very nice for governments if simply announcing that minor first offenders are not punishable would have this effect. Intuitively, it is unwanted that rules with incompatible conclusions lead to other rules, as naively as above. The property is related to the property of the so-called contraposition of the material conditional:

$$Condition \rightarrow Conclusion \vdash \neg Conclusion \rightarrow \neg Condition$$

This property can easily lead to strange results. For instance, if we have that suspects are presumed innocent, do we also have that those who are not presumed innocent are not suspect?

Both types of unwanted behavior show that rules easily allow for too many conclusions. First, we saw that a conflict of rules should not lead to a contradiction; second, that a rule should not lead to its contraposition.

This is opposite to the situation in the case of exceptions, where we saw that rules sometimes allow too few conclusions: in the standard case, we want to jump to a conclusion, even if there might be an exception.

In Figure 1, the tension between too few and too many conclusions is suggested. The set of strict conclusions that follow from a set of assumptions is often considered too small. As a result, one wants to enlarge that set by allowing tentative conclusions. On the other hand, if one enlarges the set too much, the boundary of consistency is crossed.[10] Since this is also unwanted, one wants to constrain the set of tentative conclusions, in order to maintain consistency.

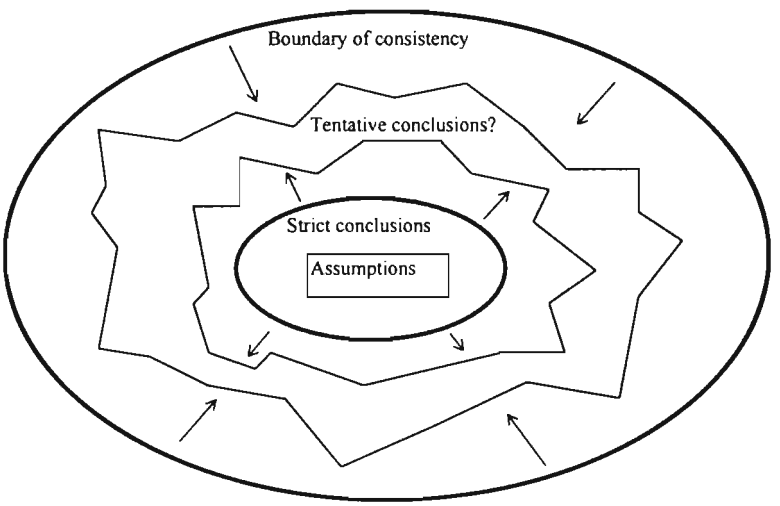

Figure 1: The tension between too few and too many conclusions

As the figure shows, an acceptable set of tentative conclusions that follow from a set of assumptions includes the set of strict conclusions, and is included in some consistent set.

Other approaches to dealing with rule conflicts are discussed in section 5.

[10] The figure suggests that there is a clear, unique, boundary of consistency. This is of course not the case: there can be many different maxiconsistent sets. However, this is unessential for what the figure attempts to depict.

2.4 Reasoning about rules

As a fourth source of problems for formalizing rules as material conditionals, we discuss reasoning about rules. We distinguish two types of reasoning about rules: reasoning with rules as conclusions, and reasoning that involves facts about rules.[11]

Assume that we consider the arguments

It is raining and I did not bring a rain coat.
So, my clothes get wet.

and

My clothes get wet.
So, I will feel uncomfortable.

to be acceptable. It seems reasonable to conclude that also the argument

It is raining and I did not bring a rain coat.
So, I will feel uncomfortable.

is acceptable. As a result, the following argument, in terms of the rules that give rise to these arguments, is acceptable:

'If it is raining and I did not bring a rain coat, my clothes get wet' is a valid rule.
'If my clothes get wet, I feel uncomfortable' is a valid rule.
So, 'If it is raining and I did not bring a rain coat, I will feel uncomfortable' is a valid rule.

This argument is an example of reasoning about rules, in which the conclusion of the argument is a rule. Other examples have facts about rules as their conclusion. There can be an argument concerning exceptions, e.g.,

John is driving on a German highway.
So, there is an exception to the rule 'If John drives faster than 120 kilometers per hour, he can be fined'.

or priority relations between rules, e.g.,

John knows Mary well.
Alex hardly knows Mary.

---

[11] We will later see (section 6) that in Reason-Based Logic this distinction disappears.

So, the rule 'If John says Mary is nice, then Mary is nice' prevails over the rule 'If Alex says Mary is not nice, then Mary is not nice' in case of a conflict.

If rules are formalized as material conditionals, the first type of reasoning about rules, in which a rule occurs as a conclusion, can apparently be dealt with. For instance, the first example we gave corresponds to the following property of material conditionals, called transitivity:

$$A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$$

However, this can hardly be considered as reasoning about rules, since it is not based on information about the particular rules involved. Transitivity is a property that holds for general material conditionals, and does not depend on any particular information for particular material conditionals.

Moreover, rules do not always have the property of transitivity. A counterexample is the following. Assume we have the two argument schemes:

Person lives in Curaçao.
So, *Person* is Dutch.

and

Person is Dutch.
So, *Person* lives in Europe.

Even if these arguments are acceptable, the argument scheme

Person lives in Curaçao.
So, *Person* lives in Europe.

need not be acceptable, since Curaçao is in the Caribbean region, and not in Europe. The fact that the property of transitivity does not hold for the arguments in this case is the result of the fact that the rule 'If someone is Dutch, he lives in Europe' can have exceptions. Since material conditionals have the property of transitivity, the rules underlying the example arguments cannot be formalized as material conditionals.

For the other type of reasoning about rules, involving facts about rules (e.g., about exceptions, conflicts or priorities), modeling rules as a material conditional is clearly inadequate, since this would require that it is possible to express facts about material conditionals in the object language. This is not possible in standard First-Order Predicate Logic.

Other approaches to dealing with reasoning about rules are discussed in section 6.

3 Relevance

In order to avoid the problems of the material conditional with regard to relevance, a special syntactic form should be used, reserved for the representation of rules. In this way, it is possible to specify the properties of rules from scratch.

We discuss three approaches that follow this idea. The first is to fixate the set of rules. The second is to treat rules as special sentences. The third is to treat rules as special objects.

3.1 Fixating a set of rules

As an example of the first type of approach, in which the set of rules is fixated, we discuss Reiter's Default Logic (Reiter, 1980, 1987). We start with a summary of his definitions.

Reiter's Default Logic uses the language of First-Order Predicate Logic; for simplicity we use that of Propositional Logic here. The assumptions are encoded as a pair of sets (F, Δ), where F is a set of sentences and Δ is a set of default rules. Such a pair of sets (F, Δ) is called a *theory*.

A *default rule* has the form

$$\alpha : \beta_1, \beta_2, \ldots, \beta_n / \gamma,$$

where α, β_1, β_2, ..., β_n, and γ are sentences. Here α is the prerequisite of the default rule, β_1, β_2, ..., β_n are the justifications of the rule, and γ is the consequent of the default rule. Representing a rule as a default rule, the condition of a rule corresponds to the prerequisite of a default rule, and the conclusion of a rule to the consequent of the default rule. The role of the justifications of a default rule is discussed in section 4.2.

An *extension* of a theory (F, Δ) is a set of sentences E, such that $E = E_0 \cup E_1 \cup E_2 \cup E_3 \cup \ldots$, where

$E_0 = F$, and
$E_{i+1} = Th(E_i) \cup \{ \gamma \mid$ there is an $\alpha : \beta_1, \beta_2, \ldots, \beta_n / \gamma \in \Delta$, such that $\alpha \in E_i$, and for all j: $\neg\beta_j \notin E \}$ for any $i \geq 0$.[12]

The definition of the E_i depends on E. Intuitively, the definition of an extension makes use of E as an advance guess of the consequences of a theory (F, Δ), and then checks whether this guess can be gradually constructed using the default rules in Δ starting from the fixed information F.[13]

---

[12] For a set of sentences S, Th(S) denotes the set of logical consequences of S in Propositional Logic.

[13] The same technique was used in the definition of the nonmonotonic consequence relation of Reason-Based Logic (chapter 2, section 6).

There is an equivalent fix-point definition of extensions: E is an extension if $E = \Gamma(E)$, where the operator Γ is defined as follows. Let S be a set of sentences. Then $\Gamma(S)$ is the smallest set Γ of sentences, such that:

$F \subseteq \Gamma$, and
$\Gamma = Th(\Gamma)$, and
For all $\alpha : \beta_1, \beta_2, ..., \beta_n / \gamma \in \Delta$: If $\alpha \in \Gamma$ and for all j: $\neg\beta_j \notin S$, then $\gamma \in \Gamma$.

(For all S, there is a smallest set with these three properties: it is the intersection of all sets for which the properties hold.)

Not all default theories (F, Δ) have an extension, and if a default theory has an extension, it is not necessarily unique. A sentence that is an element of all extensions of a default theory is said to follow *skeptically* from the theory; a sentence that is an element of (at least) one of the extensions follows *credulously*.

Reiter's starting point is the incompleteness of the information that we have about the world. He proposes to use default rules as 'rules for extending an underlying incomplete first-order theory'. Apparently, he thinks of (default) rules as special rules of inference, separate from the other available information. This is reflected in the formalism proposed. A default theory is defined as the combination of two sets: a set of first-order sentences, representing ordinary, but incomplete information about the world, and separately a set of default rules, representing information to extend the incomplete information about the world. Reiter then defines extensions of a default theory as sets of first-order sentences.

We return to our discussion of relevance. Formalizing rules as Reiter's default rules, it is clear that the problems of the material conditional with regard to relevance are solved. Since extensions cannot contain default rules, no default rule can be the consequence of a default theory. As a result, if a default rule has condition that are not considered relevant for their conclusions, it is only a flaw of the default theory.

This is of course a crude way of solving the problems of relevance. The 'advantage' is at the same time one of the main drawbacks of Reiter's Default Logic: there are no provisions whatsoever to represent relations between rules, or to reason about rules (see also section 6).

3.2 Rules as special sentences

The second approach is less crude than the first, and treats rules as special sentences. The logical language is extended with a special connective to represent rules, as in conditional logics, as for instance defined by Anderson and Belnap,[14] Nute (1980, 1994) and Delgrande (1988). After extending the language with a rule-representing connective, e.g., >, the properties of the connective are

[14] See, e.g., Haack (1978, p. 198ff.) and Sanford (1989, p. 129ff.).

specified on the meta-level by axioms and rules of inference. Some of them might be:

$A > A$
$A > C, B > C \vdash (A \vee B) > C$
$A > B, (A \wedge B) > C \vdash A > C$

The choice of such axioms and rules of inference is a delicate matter (which led to a large amount of research), and highly depends on which interpretation of rules one has in mind. For instance,

$A > B \vdash A \wedge C > B$

should hold for strict rules, but not for rules that can have exceptions.

This approach has the advantage that it is possible to represent not only rules, but also certain relations between them, namely those that can be expressed using other connectives of the logical language, as in $((A > B) \wedge (B > C)) \rightarrow (B > C)$. Of course, the axioms and rules of inference that guide this reasoning must be chosen carefully, in order to meet the demand of relevance. For instance, a rule of inference such as

$A > B \vdash A \rightarrow B$

could lead to the same unwanted results as with the material conditional, and would therefore probably be a bad choice. However, by carefully choosing axioms and rules of inference, it is in this approach in principle possible to deal with the problems of relevance.

3.3 Rules as special objects

The third approach is to treat rules as special objects, and is used in Reason-Based Logic (chapter 2). Just as in the previous approach, rules can be represented in the logical language. In Reason-Based Logic, they have the form:

rule(*condition*, *conclusion*)

However, there is an important difference with the previous approach: rules are not treated as sentences in the language, but as terms, since rules are considered as special objects. The properties of these rules-as-objects can be represented as sentences of the logical language. For instance, the validity of a rule is expressed as

Valid(rule(*condition*, *conclusion*))

RBL rules also have properties that are specified on the meta-level (described in chapter 2, section 4). For instance, an excluded rule the condition of which is satisfied is not applicable:

Condition, Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*) $\vdash\sim$
¬Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)[15]

Nevertheless, in comparison with the conditional logic approach, the properties specified on the meta-level leave much room for the specification of the rule properties in the logical language. We come back to this in section 6, where we discuss reasoning about rules.

In Reason-Based Logic this approach has been chosen, because we regard many of the properties of rules as part of the domain theory. This has the advantage that it is possible to represent different types of rules with different properties. For instance, the properties of strict rules are clearly different from those of rules that can have exceptions. In Reason-Based Logic, such properties can flexibly be represented in the domain theory. For instance, a domain theory can be such that the relevance of the rule's condition for its conclusion is implied by the rule's validity. In general, high demands are made on the domain theory.

An alternative approach to represent types of rules with different properties would be to use different syntactic structures for each type of rules. Since the properties are then represented at the meta-level (as discussed in section 3.2), this approach is a little less flexible then the approach discussed here.

4 Exceptions to rules

In this section, we discuss approaches to dealing with rules with exceptions. We do this in two parts. First, we discuss different approaches to the representation of exceptions. Second, we discuss approaches to dealing with exceptions and defeasible reasoning.

4.1 Representing exceptions

We discuss three approaches to the representation of exceptions to rules. The first uses negative rule conditions. The second uses identifiers of rules and a special predicate. The third treats rules as special objects.

[15] Recall that there is a translation from sentences (e.g., *Condition*) to terms (e.g., *condition*), as described in chapter 2, section 4.3.

Negative rule conditions

The first approach to the representation of an exception is as an additional negative condition of a material conditional, as follows:

Condition
$\wedge \neg$*Exception*
$\rightarrow$ *Conclusion*

There are two drawbacks with representing exceptions as negative conditions. The first is that an additional exception would require a change of the rule itself:

Condition
$\wedge \neg$*Exception*
$\wedge \neg$*Exception'*
$\rightarrow$ *Conclusion*

The second drawback is that there is no formal difference between the condition of a rule and its exceptions. For instance, the material conditional

$A \wedge \neg B \wedge \neg C \rightarrow D$

can represent a rule with condition A, conclusion D, and exceptions B and C, but also a rule with condition $A \wedge \neg B$, conclusion D, and exception C.

Both drawbacks conflict with the intuition that a rule is characterized by its condition and conclusion. What we would like is a system in which the existence of an additional exception to a rule is simply an additional fact about that rule.

Rule identifiers and exception predicates

The second approach to the representation of exceptions solves this disadvantage. It is characterized by the use of rule identifiers and a special purpose predicate.[16] A rule is represented as a material conditional, but has an extra condition to represent that it has no exception, for instance as follows:

(*) *Condition* $\wedge \neg$Exception(*identifier*) $\rightarrow$ *Conclusion*

Different rules should have different identifiers. Exceptions can now be represented as follows:

[16] The use of exception predicates stems from the early days of the research on nonmonotonic logics. Prakken (1993a, p. 84ff.) gives an extensive overview of different variants of this technique, in different logical formalisms.

(+) *Exception* → Exception(*identifier*)

In this representation, an additional exception does not require a change of (*), but can be represented as an additional assumption:

Exception' → Exception(*identifier*)

If such a material conditional representing an exception is itself a rule that can have exceptions, this can easily be represented by giving it its own identifier and exception clause. For instance, the material conditional (+) becomes:

Exception ∧ ¬Exception(*identifier2*) → Exception(*identifier*)

The problem with this approach to the representation of exceptions is that it is rather ad hoc. The meaning of 'rule' and 'exception' are unclear and underspecified. For instance, is a material conditional of the form (*) a rule? But then, what does the identifier of the rule refer to? Maybe the identifier is the rule? Does ¬Exception(*identifier*) imply that there is a rule with the identifier *identifier*? Taking these questions seriously, we arrive at the third approach to the representation of exceptions.

Rules as special objects

The third approach to the representation of exceptions is to treat rules as special objects that can have properties. One of the properties of a rule can be that there is an exception to the rule. So, the existence of an exception to a rule is considered as a fact about the rule. Additional exceptions do not change the rule itself, but are simply represented as additional facts about the rule.

This approach to the representation of exceptions is used in Reason-Based Logic (chapter 2). We discussed the structure of rules and several types of facts concerning rules. Rules have a condition and a conclusion:

rule(*condition*, *conclusion*)

Rules can be valid, applicable and excluded, and can apply:

Valid(rule(*condition*, *conclusion*))
Applicable(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)
Excluded(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)
Applies(rule(*condition*, *conclusion*), *fact*, *state_of_affairs*)

The general properties of rules are defined by the relations that hold between these (and other) types of facts. The properties of rules (or classes of rules) are specified in the logical language. In contrast with the previous approach using rule

identifiers, in this approach it is made explicit what is meant by 'rule' and by 'exception'.[17]

4.2 Exceptions and nonmonotonicity

Attempts to deal with nonmonotonic reasoning have become a vast field of research. Here we focus on rules with exceptions, and discuss three approaches.[18] The first is based on maxiconsistent sets. The second uses default rules. The third uses counterarguments.

Maxiconsistent sets

The first approach is based on maxiconsistent sets, as for instance used in Poole's Logical Framework for Default Reasoning (Poole, 1988).[19] We start by giving a brief overview of the definitions we need.

Poole's framework uses the language of First-Order Predicate Logic; for simplicity we use that of Propositional Logic. Assumptions are encoded in a *theory*, defined as a pair of sets (F, Δ), where F and Δ are both sets of sentences. F represents the strict assumptions, Δ the default assumptions. An *extension* of (F, Δ) is the set of consequences of a maximal *scenario*, where a scenario is a consistent set $F \cup D$ with D a subset of Δ.

A theory (F, Δ) has one or several extensions. Just as in Reiter's Default Logic (Reiter, 1980, 1987), a *credulous* and a *skeptical* consequence notion can be defined.

Maxiconsistent sets can be used to deal with reasoning with exceptions. For simplicity, we use a simple representation of rules here. A rule and its exception clause is represented as the following material conditional:

(1) *Condition* $\wedge$ ¬Exception(*identifier*) $\rightarrow$ *Conclusion*.

(Below the number (1) is used to refer to this material conditional.) Rules are elements of the strict assumptions F, and different rules should have different identifiers. As default assumptions, we have that rules have no exceptions. Formally this is achieved by including assumptions of the following form:

¬Exception(*identifier*)

---

[17] In Reason-Based Logic, there are even different ways of representing exceptions, as discussed in chapter 3, section 5.

[18] Many of the observations in this section have been made before (cf., e.g., Prakken, 1993a). See note 2.

[19] We stress that we only use the maxiconsistent sets of Poole's framework (1988) here, and *not* his way to deal with rules, described in the same paper.

in the default information Δ, for all identifiers *identifier* corresponding to a rule occurring in F. In the following, if F = $\{Ass_1, Ass_2, ..., Ass_n\}$, Δ is as above, and $Conc_1$, $Conc_2$, ..., and $Conc_m$ are elements of all extensions of the theory (F, Δ), we write:

$Ass_1, Ass_2, ..., Ass_n \vdash Conc_1, Conc_2, ..., Conc_m$

Exceptions can now be represented by an exception rule in the strict information, as follows:

(2) *Exception* $\rightarrow$ Exception(*identifier*).

If there is no exception, the default assumption that there is no exception does not lead to a contradiction, so we have

Condition, (1), (2) $\vdash$ *Conclusion*, $\neg$Exception(*identifier*)

In the case of an exception, the exception rule (2) gives the following:

Condition, *Exception*, (1), (2) $\vdash$ Exception(*identifier*)

Since $\neg$Exception(*identifier*) does not follow, the conclusion of the rule does not follow, in other words:

Condition, *Exception*, (1), (2) $\nvdash$ *Conclusion*

Since this corresponds to the two intuitive requirements STANDARD CASE and EXCEPTION CASE discussed in section 2.2, everything seems to work out fine.

However, a problem arises if there are exceptions to the exception rule itself. Exceptions to exceptions are a common phenomenon. In such a case the conclusion of the rule should follow in spite of the exception. We add a third intuitive requirement

EXCEPTION-TO-EXCEPTION CASE

If there is a rule the condition of which holds, there is an exception to the rule, and there is an exception to the exception, then the rule's conclusion follows.

In order to meet this requirement, we need to represent exceptions to the exception rule. Therefore the exception rule (2) above is replaced by the following rule, that can have exceptions:

(3) *Exception* $\wedge$ $\neg$Exception(*identifier2*) $\rightarrow$ Exception(*identifier*).

Furthermore we have an exception to the exception:

(4) *Exception_to_exception* → Exception(*identifier2*).

Since there is only one extension containing ¬Exception(*identifier*) and Exception(*identifier2*), we obtain the correct behavior in case of an exception to an exception:

Condition, *Exception*, *Exception_to_exception*, (1), (3), (4) $\vdash_{\sim}$ *Conclusion*

Unexpectedly, we have lost the correct behavior in the exception case: the theory (F, Δ) with F = {*Condition*, *Exception*, (1), (3), (4)} and Δ as above has two extensions. One extension contains both Exception(*identifier*) and ¬Exception(*identifier2*), as desired. The other contains ¬Exception(*identifier*), and *Conclusion*, but remains silent about whether the exception rule is applicable: it contains neither ¬Exception(*identifier2*) nor Exception(*identifier2*). It should be noted that in an extension containing ¬Exception(*identifier*) the inclusion of ¬Exception(*identifier2*) is blocked since that would give an inconsistency with (3). The exception rule (3) just demands that an extension that contains *Exception*, can only contain one of the sentences ¬Exception(*identifier*) and ¬Exception(*identifier2*). This demand is met in both extensions.

What is wrong is that the second extension does not contain the fact that there is no exception to the rule with identifier *identifier2*, only in order to maintain consistency. The first extension contains the fact that there is an exception to the rule *identifier* because there is an exception. Intuitively, we want that the application of a rule can only be blocked by explicit information in the extension.

Default rules

The second approach that we discuss uses Reiter's (1980, 1987) default rules. We use the definitions discussed in section 3.1. Default rules have, apart from a condition and a conclusion, a justification. This justification is used to block the application of a rule in case it follows that there is an exception. We do not have to assume by default that there is no exception to a rule, as in the previous approach.

Rules are represented as default rules as follows:

(5) *Condition* : ¬Exception(*identifier*) / *Conclusion*

Again it is assumed that different rules have different identifiers. An exception rule is represented as:

(6) *Exception* : ¬Exception(*identifier2*) / Exception(*identifier*)

An exception-to-exception rule is represented as:

(7) *Exception_to_exception* : ¬Exception(*identifier3*) / Exception(*identifier2*)

It turns out that the representation of rules and exceptions in this way meets the requirements, including that of the exception-to-exception case. To see the difference with the maxiconsistent set approach, we look what happens in the exception case that was problematic there.

We start with the default theory (F, Δ), where F = {*Condition*, *Exception*} and Δ = {(5), (6), (7)}. We propose two sets of sentences E and E* as guesses for extensions. They correspond to the two extensions in the maxiconsistent set approach:

$E = Th(F \cup \{Exception(identifier)\})$[20]
$E^* = Th(F \cup \{\neg Exception(identifier2), Conclusion\})$

The set E is indeed an extension, since we have:

$E_0 = F$, and
$E_1 = Th(F \cup \{Exception(identifier)\}) = E$, and
$E_i = E_1$, for all $i > 1$.

and therefore $E = \bigcup_i E_i$, as required. But the set E* is not an extension. We have:

$E^*_0 = F$, and
$E^*_1 = Th(F \cup \{Conclusion\})$, and
$E^*_i = E^*_1$, for all $i > 1$.

As a result $\bigcup_i E^*_i = E^*_1$, which is a proper subset of E*. Since no information in the assumptions supports that there is no exception to the rule *identifier2*, the sentence ¬Exception(*identifier2*) cannot be an element of an extension.

We conclude that this approach using default rules can adequately deal with the three requirements for reasoning with rules with exceptions.

Counterarguments

The third approach that we discuss uses counterarguments. We base the discussion here on Pollock's Theory of Defeasible Reasoning (1987-1995). We start by giving a description of some of his definitions, adapted to suit our needs.

An *argumentation theory* is a pair of sets (*Args*, *Defs*), such that *Defs* is a set of pairs of elements of *Args*. The elements of *Args* are called *arguments*, the elements of *Defs defeaters*. If (α, β) is an element of *Defs*, the argument α is said to *defeat* the argument β. Pollock then defines *levels*, as follows:

---

20 Here Th(S) denotes the *deductive closure* of S, i.e., the set of all deductive consequences of S.

- All arguments are *in at level 0.*
- An argument is *in at level n + 1* if and only if it is in at level 0 and it is not defeated by any argument that is in at level n.

An argument is *ultimately undefeated* if and only if there is a level such that it is in at that level and at all higher levels. An argument is *ultimately defeated* if and only if there is a level such that it is out at all higher levels. An argument is *provisionally defeated* if and only if it is neither ultimately undefeated nor ultimately defeated.

Pollock's Theory of Defeasible Reasoning can be used to represent reasoning with rules with exceptions as follows.[21] We define a *theory of reasoning* as a pair of sets (*Facts*, *Rules*), where *Facts* are elements of some language L and Rules have the form *Condition* → *Conclusion*, where *Condition* and *Conclusion* are elements of the language. It is assumed that the language L contains identifiers for the rules in Rules, and has a predicate to represent exceptions. For instance, the fact that there is an exception to the rule *Condition* → *Conclusion* with identifier *id*, might be expressed as follows:

Exception(*id*)

For a theory of reasoning (*Facts*, *Rules*), we can define an argumentation theory (*Args*, *Defs*), as follows. The set *Args* consists of all facts and all loop-free chains of rules starting from the facts. The set *Defs* consists of pairs of arguments (α, β), such that the argument α ends with Exception(*id*), where *id* is the identifier of a rule in the argument β.

As an example, we assume that the set Rules consists of the following three rules, with identifiers *id1*, *id2* and *id3*, respectively:

Condition → *Conclusion*
Exception → Exception(*id1*)
Exception_to_exception → Exception(*id2*)

We discuss what happens in the standard, the exception and the exception-to-exception case. In the standard case, the set of facts only contains *Condition*. In that case, the only arguments are *Condition* and *Condition* → *Conclusion*, and there is no defeater. As a result, both arguments are in at all levels, and are ultimately undefeated.

In the exception case, the set of facts consists of *Condition* and *Exception*. There are two new arguments, namely *Exception* and *Exception* → Exception(*id1*). Now there is one defeater, namely (*Exception* → Exception(*id1*), *Condition* → *Conclusion*). We have:

[21] Here we do not follow Pollock.

The arguments *Condition*, *Condition* → *Conclusion*, *Exception* and *Exception* → Exception(*id1*) are in at level 0.
The arguments *Condition*, *Exception* and *Exception* → Exception(*id1*) are in at level 1, and at all higher levels.

So, the arguments *Condition*, *Exception* and *Exception* → Exception(*id1*) are ultimately undefeated, and the argument *Condition* → *Conclusion* is ultimately defeated. The latter argument is of course defeated by the argument *Exception* → Exception(*id1*).

In the exception-to-exception case, the set of facts consists of *Condition*, *Exception* and *Exception_to_exception*. The new arguments are *Exception_to_exception* and *Exception_to_exception* → Exception(*id2*). The new defeater is (*Exception_to_exception* → Exception(*id2*), *Exception* → Exception(*id1*)). We have:

The arguments *Condition*, *Condition* → *Conclusion*, *Exception*, *Exception* → Exception(*id1*), *Exception_to_exception* and *Exception_to_exception* → Exception(*id2*) are in at level 0.
The arguments *Condition*, *Exception*, *Exception_to_exception* and *Exception_to_exception* → Exception(*id2*) are in at level 1.
The arguments *Condition*, *Condition* → *Conclusion*, *Exception*, *Exception_to_exception* and *Exception_to_exception* → Exception(*id2*) are in at level 2, and at all higher levels.

So, all arguments are ultimately undefeated, except the argument *Exception* → Exception(*id1*), that is ultimately defeated. The latter argument is defeated by the argument *Exception_to_exception* → Exception(*id2*).

5 Rule conflicts

In this section, we discuss approaches to dealing with rule conflicts. We do this in two parts. First, we discuss different approaches to the representation of conflict resolving information. Second, we discuss approaches to dealing with conflicts and consistency maintenance.

5.1 Representing conflict resolving information

We start with a discussion of approaches to the representation of conflict resolving information. We distinguish three types of conflicts: conflicts of pairs of rules, bipolar multiple conflicts, and general multiple conflicts. For each type of conflict, we discuss a corresponding type of conflict resolving information: rule priorities, weighing, and general conflict resolution, respectively.

Conflicts of pairs of rules and rule priorities

The simplest, and most common, type of rule conflict is the conflict of two rules: there are two rules with opposite conclusions and the conditions of both rules are satisfied.

If there is a conflict of a pair of rules, often one of the rules prevails over the other. We have seen several examples in chapter 3, section 6.1. As a result of such priority information, the conflict is resolved. The prevailing rule leads to its conclusion, while the other rule does not. Clearly, the conflict of rules leads to a special type of exception to the non-prevailing rule. As a result, rule priorities can be represented using the techniques already discussed in section 4.1 on representing exceptions.

Assume that we have two rules with incompatible conclusions represented as the following two material conditionals:

$$Condition_1 \wedge \neg\text{Exception}(identifier_1) \rightarrow Conclusion$$
$$Condition_2 \wedge \neg\text{Exception}(identifier_2) \rightarrow \neg Conclusion$$

Assume moreover that the first prevails over the other. This priority information can now be represented as follows:

$$Condition_1 \wedge \neg\text{Exception}(identifier_1) \rightarrow \text{Exception}(identifier_2)$$

Abbreviating $Condition_i \wedge \neg\text{Exception}(identifier_i)$ as $\text{Applicable}(identifier_i)$ (for i = 1 or 2), we obtain the following sentence:

$$\text{Applicable}(identifier_1) \rightarrow \text{Exception}(identifier_2)$$

It may be tempting to represent the priority information as the following sentence:

$$\text{Applicable}(identifier_1) \rightarrow \neg\text{Applicable}(identifier_2)$$

However, this is an incorrect representation, since this sentence is symmetric in the two rules, as its equivalent

$$\neg\text{Applicable}(identifier_1) \vee \neg\text{Applicable}(identifier_2)$$

clearly shows.

Bipolar multiple conflicts and weighing

The second type of rule conflict that we discuss are bipolar multiple conflicts: two groups of rules have equal conclusions in each group, but incompatible conclusions across the groups, while the conditions of the rules are satisfied.

For instance, the following material conditionals represent a bipolar conflict of a group of n rules and a group of m rules (where n and m are natural numbers):

$Condition_{11} \wedge \neg\text{Exception}(identifier_{11}) \rightarrow Conclusion$
...
$Condition_{1n} \wedge \neg\text{Exception}(identifier_{1n}) \rightarrow Conclusion$
$Condition_{21} \wedge \neg\text{Exception}(identifier_{21}) \rightarrow \neg Conclusion$
...
$Condition_{2m} \wedge \neg\text{Exception}(identifier_{2m}) \rightarrow \neg Conclusion$

We have seen examples in which such a conflict cannot be resolved by priority information on pairs of rules, but by priority information on *groups* of rules (chapter 2, section 1.3; chapter 3, section 4).

The priority technique used for pairwise conflicts can be extended to the case of bipolar multiple conflicts. For instance, if the first group of n rules above prevails over the second group of m rules, this can be represented as follows:

$\text{Applicable}(identifier_{11}) \wedge ... \wedge \text{Applicable}(identifier_{1n}) \rightarrow \text{Exception}(identifier_{21}) \wedge$
$... \wedge \text{Exception}(identifier_{2m})$

In Reason-Based Logic (chapter 2), a representation similar to this one is possible. However, Reason-Based Logic provides a second way of representation, using the weighing of reasons. The priority of the first group of rules over the second is represented as the fact that the reasons that result from the first group of rules outweigh the reasons from the second group:

$\text{Outweighs}(\{condition_{11}, ..., condition_{1n}\},$
$\{condition_{21}, ..., condition_{2m}\},$
$conclusion)$

The two techniques seem to lead to similar results. However, there is a technical difference. The two expressions representing conflict resolving information are not equivalent, because the weighing expression only helps to resolve the conflict if there is no other rule with conclusion $\neg conclusion$ (cf. the relations between facts described in chapter 2, section 5), while the generalized priority expression helps to resolve the conflict also in that case. The use of the weighing expression reflects the intuition that the bipolar multiple conflict should only be resolved if all rules of the losing side, i.e., those with conclusion $\neg conclusion$, have been considered. In the more familiar terminology of reasons, the weighing information only should have effect if all counterreasons have been considered.

As a result, the explicit representation of the weighing of reasons as in Reason-Based Logic seems to be closer to the examples of accrual of reasons, that led to the distinction of bipolar rule conflicts.

General multiple conflicts and general conflict resolution

As a third type of rule conflict, we discuss general rule conflicts: there is a group of rules with incompatible conclusions, the conditions of which are satisfied. For instance, we might have:

$Condition_1 \wedge \neg\text{Exception}(identifier_1) \rightarrow Conclusion_1$
...
$Condition_n \wedge \neg\text{Exception}(identifier_n) \rightarrow Conclusion_n$
$\neg(Conclusion_1 \wedge ... \wedge Conclusion_n)$

We have seen two special cases of resolutions of such a general rule conflict:

1. One of the rules might prevail over another.
2. A subgroup of rules might prevail over another subgroup of rules with incompatible conclusion.

The most general type of conflict resolution would require the representation of the prevalence of *any* subgroup over *any* other subgroup, formally:

$\text{Prevails}(\{identifier_{11}, ..., identifier_{1n}\}, \{identifier_{21}, ..., identifier_{2m}\})$

We do not know a formalism in which this is explicitly done, although it is a natural generalization of the two discussed representation techniques, i.e., using exceptions and using weighing, to the case of general multiple conflicts.

5.2 Conflicts and consistency maintenance

Since there is not always sufficient information to resolve rule conflicts, many techniques have been proposed to prevent the unwanted effects of contradiction by means of consistency maintenance. Here we discuss three such techniques. We start with Reiter's normal and semi-normal default rules (Reiter, 1980, 1987), then we discuss Vreeswijk's use of conclusive force (Vreeswijk, 1991, 1993), and we finish with Pollock's collective defeat (Pollock, 1987).

Normal and semi-normal default rules

The first approach to consistency maintenance in case of rule conflicts that we discuss are the normal and semi-normal default rules of Reiter's Default Logic (Reiter, 1980, 1987). In section 4.2, we already discussed how default rules (Reiter, 1980, 1987) can be used to represent rules with exceptions. There, a rule was represented as a default of the following form:

Condition : ¬Exception(*identifier*) / *Conclusion*

If two default rules of this form are in conflict, there is no extension. An example is the theory (F, Δ) defined as follows:

F = {*Condition*$_1$, *Condition*$_2$}
Δ = {*Condition*$_1$: ¬Exception(*identifier*$_1$) / *Conclusion*,
Condition$_2$: ¬Exception(*identifier*$_2$) / ¬*Conclusion*}

As a result, using the skeptical consequence relation of Default Logic, everything follows from such a theory. This behavior resembles the behavior of an inconsistency in classical logic.

There is another type of default rule that can never give rise to this behavior: default rules of this type are called normal default rules, and have the form

Condition : *Conclusion* / Conclusion.

Informally, a default rule leads to its conclusion if its condition is satisfied, unless that would lead to an inconsistency. Normal defaults have the nice formal property that a theory that only contains normal default rules always has an extension.

Reiter (1980, 1987) claimed that normal default rules were sufficient in practice. However, as was already noted by Reiter and Criscuolo (1981, 1987), normal default rules are not always sufficient. We saw above that non-normal default rules are needed to represent rules with exceptions.

In order to catch the benefits of both, a combined form can be used, as follows:

Condition : ¬Exception(*identifier*), *Conclusion* / *Conclusion*

Default rules that have their conclusion as one of their justifications are called semi-normal. Informally, a default rule of this form leads to its conclusion if its condition is satisfied, unless Exception(*identifier*) or ¬*Conclusion* would also follow.

Two conflicting rules will now give rise to two extensions. For instance, the theory (F, Δ) with

F = {*Condition*$_1$, *Condition*$_2$}
Δ = {*Condition*$_1$: ¬Exception(*identifier*$_1$), *Conclusion* / *Conclusion*,
Condition$_2$: ¬Exception(*identifier*$_2$), ¬*Conclusion* / ¬*Conclusion*}

has two extensions E_1 and E_2:

E_1 = Th({*Condition*$_1$, *Condition*$_2$, *Conclusion*})
E_2 = Th({*Condition*$_1$, *Condition*$_2$, ¬*Conclusion*})

In each extension, only one of the rules has led to its conclusion. Intuitively, the two extensions can arise because there are two orders in which the defaults can be used: first drawing the conclusion of rule *identifier*$_1$ blocks using rule *identifier*$_2$, while first drawing the conclusion of *identifier*$_2$ blocks using rule *identifier*$_1$.

However, a theory with only semi-normal default rules does not always have an extension, as the theory (F, Δ) with

$F = \{Condition_1, Condition_2, Condition_3\}$
$\Delta = \{Condition_1 : \neg\text{Exception}(id_1), \text{Exception}(id_2) / \text{Exception}(id_2),$
$Condition_2 : \neg\text{Exception}(id_2), \text{Exception}(id_3) / \text{Exception}(id_3),$
$Condition_3 : \neg\text{Exception}(id_3), \text{Exception}(id_1) / \text{Exception}(id_1)\}$

shows.

Conclusive force

The second approach to consistency maintenance in case of conflicts that we discuss is Vreeswijk's use of the conclusive force of arguments. We give a simplified overview of some definitions of Vreeswijk's (1991, 1993) Abstract Argumentation Systems.

Vreeswijk starts with the definition of an *argumentation system* as a triple (*Language*, *Rules*, <). Here *Language* is any set containing a special element ⊥, denoting contradiction. This set is called the *language* of the argumentation system. The set *Rules* is a set of rules, that have the form *Condition*$_1$, ..., *Condition*$_n$ → *Conclusion*. The *conclusive force relation* < is a strict order on arguments, that are tree-like chains of rules.

He proceeds with the definition of defeasible entailment and extensions, which uses the notion of conflict. A set of arguments *Arguments* is *in conflict with* an argument *Argument* (relative to a set *Assumptions* ⊆ *Language*), if *Argument* and elements of *Arguments* are parts of a larger argument with conclusion ⊥ and with premises in the set *Assumptions*. A relation $\vdash$ between sets of sentences of the language and arguments is called a *defeasible entailment relation* if the following holds for all sets *Facts* ⊆ *Language* and arguments *Argument*:

Assumptions $\vdash$ *Argument* if and only if one of the following holds:

1. *Argument* is an element of *Assumptions*
2. *Argument* has the form *Argument*$_1$, ..., *Argument*$_2$ → *Conclusion*, and for every set of arguments *Arguments*, such that *Assumptions* $\vdash$ *Argument'* for all elements *Argument'* of *Arguments*, we have:
   If *Arguments* is in conflict with *Argument* (relative to *Assumptions*), then there is a *Argument'* in *Arguments*, such that *Argument'* < *Argument*.

(This is not a definition of the relation $\vdash$ by recursion on arguments, since $\vdash$ appears on both sides of the 'if and only if'. Such a definition would be unexpected since $\vdash$ is nonmonotonic.) An *extension* of a set *Assumptions* is then defined as a set of arguments *Arguments*, such that *Arguments* = {*Argument* | *Assumptions* $\vdash$ *Argument*}.

How can Vreeswijk's formalism be used to maintain consistency in case of rule conflicts? In Vreeswijk's formalism, conflicts of rules occur as conflicts of the final steps of arguments. Informally, Vreeswijk's definition has the result that such conflicts between arguments are resolved by 'throwing away' one argument that is involved in the conflict.

To choose an argument, the conclusive force relation is used: an argument cannot be thrown away if it is stronger than any of the other arguments involved in the conflict. Since there can still remain more than one argument that can be chosen, multiple extensions can arise.

Collective defeat

As a third approach to consistency maintenance in case of rule conflicts, we mention Pollock's collective defeat (Pollock, 1987).

He proposes to withhold from drawing a conclusion in case there is an unresolved conflict of rules. He achieves this by considering all arguments with conflicting last steps as defeated in case of an unresolved conflict: the arguments are *collectively defeated.*

In Reason-Based Logic (chapter 2), there is a variant of collective defeat: if there are conflicting reasons, but there is no weighing information available, no conclusion follows. As a result, while Pollock's collective defeat can maintain consistency for general multiple rule conflicts, Reason-Based Logic uses a form of collective defeat in the specific case of bipolar multiple conflicts.

6 Reasoning about rules

Below, we discuss three approaches to dealing with reasoning about rules. The first is to treat rules as special sentences. The second is to use rule identifiers. The third is to treat rules as special objects.

6.1 Rules as special sentences

The first approach to dealing with reasoning about rules is to treat rules as special sentences, as in conditional logics (e.g., Nute, 1980, 1994; Delgrande, 1988). In conditional logics, the properties of a special connective are specified on the meta-level. We already discussed the conditional logic approach in section 3.2 on relevance. There, we mentioned conditional logics because they make it possible to

take the requirement of relevance into account. But conditional logics also are regarded as logics that can treat reasoning with rules.

For instance, assume we want to represent a transitive type of rule. Then one of the defining properties of the rule-representing conditional would be:

$A > B, B > C \mid\sim A > C$

This rule of inference makes it possible to derive the conditional $A > C$ from the conditionals $A > B$ and $B > C$.

If we wish to represent a type of rule that is not transitive, one of the defining properties of the conditional representing the rule type, might have the following weaker property:

$A > B, (A \wedge B) > C \mid\sim A > C$

By choosing the defining properties, we can specify different forms of reasoning about rules.

However, there are two limitations. The first limitation is that in this way it is impossible to distinguish classes of rules unless each class of rules is represented by a syntactically different conditional. As a result, properties of rules of different kinds, such as transitive and intransitive rules, can only be represented at the meta-level, and not, more flexibly, at the logical level.

The second limitation is that in conditional logics, it is impossible to represent facts about rules, other than that they are valid or invalid. As a result, although it is possible to represent the first type of reasoning about rules distinguished in section 2.4, i.e., reasoning with rules as conclusions, it is not possible to represent the second type, i.e., reasoning with facts about rules.

6.2 Rule identifiers

The second approach to dealing with reasoning about rules is an attempt to deal with these limitations, and is the technique of rule identifiers, already discussed as one of the techniques to represent exceptions in section 4.1.

This technique can be used in a more general way to deal with reasoning about rules. To represent exceptions to rules the rule identifiers were only used in the special purpose predicate Exception(*identifier*). However, rule identifiers can also be used as parameters for other predicates. For instance, the conclusion of a priority argument says that some rule prevails over another rule. If the identifiers of these rules are $identifier_1$ and $identifier_2$, this can be represented as follows:

Prevails($identifier_1$, $identifier_2$)

As a result, if this technique is used, it is possible to represent the second type of reasoning about rules distinguished in section 2.4, i.e., reasoning with facts about

rules, but it is not clear how to represent the first type, i.e., reasoning with rules as conclusions. This limitation is the result of the fact that the approach is unclear about the status of rules and identifiers, as was already noted in section 4.1 on representing exceptions.

6.3 Rules as special objects

The latter brings us to the third approach to dealing with reasoning about rules, namely to consider rules as special objects, as in Reason-Based Logic (chapter 2). This approach was also discussed in section 3.3 on relevance and section 4.1 on representing exceptions.

As we will see, this approach can be regarded as an integration of the two other approaches, keeping the benefits of both. To recall, rules are treated as objects, that are represented as follows:

rule(*condition*, *conclusion*)

Transitivity of rules can now be represented as

Valid(rule(*a*, *b*)), Valid(rule(*b*, *c*)) $\vdash$ Valid(rule(*a*, *c*))

Transitivity of rules can be restricted to a certain class of rules by explicitly mentioning such a class, here named transitive_class:

Class(transitive_class, rule(*a*, *b*)), Class(transitive_class, rule(*b*, *c*)),
Valid(rule(*a*, *b*)), Valid(rule(*b*, *c*)) $\vdash$ Valid(rule(*a*, *c*))

Facts about rules are stated using the complete rule, instead of using only an identifier. The former is more expressive. For instance, whereas for the representation of an exception to a rule it is sufficient to use an identifier, as in

Excluded(*identifier*),

for the representation of the validity of a rule the complete rule, including its condition and conclusion are needed, as in the following sentence:

Valid(rule(*condition*, *conclusion*))

As explained in chapter 2, section 3.1, this approach requires a translation from sentences to terms. This translation is already necessary in order to draw a conclusion from a valid rule, as in ordinary rule application:

Valid(rule(*condition*, *conclusion*)), *Condition* $\vdash$ *Conclusion*

For details on the formal definition of such a translation, the reader is referred to chapter 2, section 4.3.

In this approach, the properties of rules can be specified both on the meta-level and on the level of the logical language. As a result, the meta-level can be used to specify the general properties of rules that are considered most basic, such as rule application and its relation to exceptions, while the level of the logical language can be used to specify the specific properties of specific rules in a specific case or domain.

Therefore, this approach can deal with both types of reasoning with rules that were distinguished in section 2.4, in contrast with the conditionals and identifiers approach.

Chapter 5

CumulA: a model of argumentation in stages

The previous chapters dealt with the nature of the rules and reasons that are at the basis of argumentation. In this chapter, we investigate the process of argumentation itself. We focus on arguments and their defeat. This leads to a formal model of argumentation in stages, called CumulA.[1]

In section 1, we introduce argumentation with defeasible arguments and give an overview of CumulA. In section 2, arguments and their structure are treated. In section 3, we discuss how the defeat of arguments is formalized using defeaters. In section 4, the stages of argumentation are characterized. Section 5 deals with lines of argumentation and argumentation diagrams. Section 6 gives a number of examples.

1 Argumentation in stages

Below, we first give an informal introduction of the key terminology, related to arguments and defeat as it is used in this chapter. Second, we give an overview of the formal model CumulA.

1.1 Arguments and defeat

The goal of argumentation is to find (rationally) *justified* conclusions (cf., e.g., Pollock, 1987). For instance, if a colleague enters the room completely soaked and tells that it is raining outside, I would of course conclude that it is wise to put on a raincoat. My conclusion is rationally justified, since I can give *support* for it, namely the fact that my colleague is completely soaked and tells me that it is raining. If I were asked why I concluded that it is wise to put on a raincoat, I could answer with the following *argument*:

---

[1] The name CumulA is an abbreviation of Cumulative Argumentation, but was chosen since it reminds of a certain type of cloud, the cumulus. The formal model is based on

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

Such an argument is a reconstruction of how a conclusion can be supported. The argument given here consists of two *steps*. In general, an argument can support its conclusion if the steps in the argument are based on *rules* (see also chapter 4, section 1). Here we do not answer the question which argument steps can give rise to arguments that support their conclusion and which do not, or, in other words, which steps are based on rules and which are not. We assume that the rules allowing argument steps are somehow given.[2]

An argument that supports its conclusion does not always justify it. For instance, if in our example I look out the window and see wet streets, but otherwise a completely blue sky, I would conclude that the brief shower is over. So, while at the time my colleague entered it was justified for me to conclude that it is wise to put on a raincoat, it is not justified anymore after looking out the window. In this case, we say that the argument is *defeated*. In the example, the argument

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

does not justify its conclusion because of the argument

The streets are wet, but the sky is completely blue.
So, the shower is over.

In this case the argument that it is probably raining is defeated by the argument that the shower is over. The new information that the shower is over has the effect that the argument does not justify its conclusion, but does not change the fact that in principle the argument supports its conclusion.[3]

Our example has illustrated two points about argumentation, that form the basis of our model:

1. Argumentation is a process (see also chapter 1, section 1), in which at each stage new arguments are taken into account.

---

previous work (Verheij, 1995a, b, c). However, most definitions are new or have been changed considerably.

2 We believe that in the end the rules and reasons on which argument steps are based are a special kind of *memes* (cf. Dawkins, 1989). An interesting account of the relation between rationality and evolution is given by Rescher (1988, p. 176ff.).

3 It should be recalled that in our use of terminology justification of a conclusion does not imply truth of the conclusion (cf. chapter 1, section 1).

2. Each argument that is taken into account has either one of two statuses: the argument is either undefeated or defeated, indicating that the argument justifies its conclusion, or not, respectively.[4]

Our example showed that the status of an argument can depend on the structure of the argument, the counterarguments that are taken into account, and the argumentation stage.

- *The structure of the argument*
  The two-step argument that it is wise to put on a raincoat is defeated because already its first step, in which it is concluded that it is probably raining, is defeated.

- *Counterarguments*
  The argument that the shower is over defeats the argument that it is probably raining.

- *The argumentation stage*
  The argument that it is probably raining is only defeated once the argument that the shower is over has been taken into account.

1.2 Overview of CumulA

In this chapter, a formal model of argumentation with defeasible arguments is developed. This model is called CumulA. Formally, it builds on Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993), Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993), and Dung's Argumentation Frameworks (Dung, 1993, 1995). Key definitions in CumulA are those of *arguments*, *defeaters*, *argumentation theories*, *stages* and *lines of argumentation*.

- *Arguments*
  Arguments in CumulA are tree-like structures that represent how a conclusion is supported. Arguments are the subject of section 2. Our composite arguments are not, as usual, only constructed by subordination, but also by coordination. For instance, Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993) use subordination, but not coordination in their argumentation models. We investigate how the coordination of arguments is related to defeat.

---

[4] Cf. Pollock (1987-1995) and Vreeswijk (1991, 1993). Prakken (1993a, b) considers a third status: an argument can be defensible arguments, which means that it is neither undefeated nor defeated.

- *Defeaters*
  Defeaters indicate which arguments can defeat which other arguments. They consist of a set of challenging arguments and a set of challenged arguments. If the challenging arguments of a defeater are undefeated, they defeat its challenged arguments.[5] Defeaters are treated in section 3. We show that our defeaters can represent a wide range of types of defeat. Our defeaters are formally related to Dung's (1995) attacks. However, our defeaters can represent how the structure of arguments is related to defeat, and can represent more general types of defeat in which groups of arguments challenge other groups of arguments.

- *Argumentation theories*
  Argumentation depends on the language that is used, on the arguments that can support conclusions, and on which arguments defeat which other arguments. This information is represented as an argumentation theory. An argumentation theory consists of a set of sentences (together forming the language), a set of rules that give rise to arguments, and a set of defeater schemes that determine which arguments defeat which other arguments. In order to define forward and backward lines of argumentation (see below), an argumentation theory does not fix the premises of the arguments, as for instance in Lin and Shoham's (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk's (1991, 1993) argumentation models. Argumentation theories are characterized at the end of section 3.

- *Stages*
  A stage in the argumentation process is characterized by the arguments that have been taken into account, and by the defeat status of these arguments, either undefeated or defeated. Which stages are allowed is determined by an argumentation theory. A stage consists of a pair of sets, one of them representing the arguments that are undefeated at the stage, the other the arguments that are defeated at the stage. The union of these sets represents which arguments have been taken into account. Stages are discussed in section 4. Vreeswijk's (1991, 1993) argument structures are comparable to our stages. However, they are representations of the arguments currently undefeated, and not of all arguments currently taken into account, whether undefeated or defeated.

- *Lines of argumentation*
  Argumentation can proceed in many ways, depending on the obtaining goals, protocols and strategies of argumentation. This leads to different lines of argumentation. A line of argumentation is a sequence of consecutive

---

5 Our notion of defeat is *counterargument-triggered*, as, e.g., Dung's (1995), and not *inconsistency-triggered*, as, e.g., Vreeswijk's (1991, 1993). We discuss this distinction more extensively in chapter 6, section 4.

argumentation stages. In a line of argumentation, arguments are gradually constructed. At each stage in a line of argumentation, the arguments taken into account have a defeat status. However, the status of an argument can change during a line of argumentation. Which lines of argumentation are possible is determined by an argumentation theory. We also define argumentation diagrams that represent several possible lines of argumentation as allowed by an argumentation theory. Lines of argumentation and argumentation diagrams are defined in section 5. Our lines of argumentation are related to Vreeswijk's (1991, 1993) argumentation sequences. However, since our argumentation theories do not fix the allowed premises, in our lines of argumentation arguments can be constructed not only forwardly, but also backwardly.

We have to make a disclaimer here: our definition of lines of argumentation does not prescribe how argumentation should proceed, but only attempts to describe which lines of argumentation are possible.[6] We return to this issue in section 5.3.

2 Arguments and their structure

This section deals with the structure of arguments. We treat arguments as tree-like structures of sentences, similar in form to logical proofs. After an informal discussion of elementary and composite argument structures (sections 2.1 and 2.2), we give a formal definition of arguments in section 2.3. The section ends with the definition of initials and narrowings of arguments (section 2.4).

2.1 Elementary argument structures

The simplest type of argument is the *statement*. Examples of statements are:

The sky is blue.

and

The film was good.

In principle, any (assertive) sentence can be used as a statement, so schematically statements have the following trivial structure:

Sentence.

---

6 Recently several protocols prescribing (or at least constraining) lines of argumentation have been proposed, especially in a dialogical setting (see, e.g., Gordon, 1993a, 1993b, 1995; Brewka, 1994; Lodder and Herczog, 1995).

Some would hesitate to call statements arguments, because of their trivial structure. Since statements can be considered as the beginning of all argumentation, it will turn out convenient to include statements in the definition of arguments.

The simplest type of argument with non-trivial structure is the *single-step argument*, for instance:

The sun is shining.
So, it is a beautiful day.

Schematically, a single-step argument has the following structure:

Reason.
So, *Conclusion.*

A reason in an argument can consist of a several subreasons, as for instance in the following argument that has two subreasons:

Alex has an appointment at eight with John in Maastricht, John has an appointment at seven with Mary in Amsterdam.
So, John cannot keep both appointments.

Schematically, we have:

Subreason$_1$, *Subreason*$_2$, ..., *Subreason*$_n$.
So, *Conclusion.*

We use different terms 'subreason' and 'reason', since only the combination of the subreasons provides a reason that supports the conclusion. It should be noted that this is in contrast with everyday language, where the distinction between subreasons and reasons is not made, and both are called reasons.

2.2 Composite argument structures

Arguments can be combined. There are two basic ways to combine arguments into more complex structures, namely subordination and coordination.[7]

- Subordination of arguments

  If a single-step argument has a conclusion that is the same as one of the reasons or subreasons of another argument, arguments can be subordinated. We have already seen an example of subordination, namely:

[7] We use the same argument structure as Van Eemeren *et al.* (1981, 1987), but our terminology is different. Our coordinated arguments correspond to their multiple arguments.

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

This argument is the result from the subordination of the arguments

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

and

It is probably raining.
So, it is wise to put on a raincoat.

Schematically,

Reason.
So, *Conclusion*$_1$.
So, *Conclusion*$_2$.

- Coordination of arguments

If two arguments have the same conclusion, they can be coordinated. For instance, if the arguments

The sun is shining.
So, it is a beautiful day.

and

The sky is blue.
So, it is a beautiful day.

are coordinated, we obtain

The sun is shining; The sky is blue.
So, it is a beautiful day.

Schematically, we have

Reason$_1$; *Reason*$_2$.
So. *Conclusion.*

It should be noted that, in contrast with the subreasons mentioned earlier, each reason in a coordinated argument supports the conclusion on its own. To distinguish reasons and subreasons, reasons are separated by semicolons, while subreasons are separated by commas.[8] For instance, an argument can have the following structure:

$Subreason_{11}$, $Subreason_{12}$; $Subreason_{21}$, $Subreason_{22}$.
So, *Conclusion*.

Here $Subreason_{11}$ and $Subreason_{12}$ together form a reason for the conclusion, while $Subreason_{21}$ and $Subreason_{22}$ form a separate, second reason for it.

By repeating these two ways of combining arguments, the structure of arguments can become arbitrarily complex.

2.3 Definition of arguments

In our model of argumentation, we abstract from the language, and therefore treat a language simply as a set without any structure. Here we follow Lin and Shoham (Lin and Shoham, 1989; Lin, 1993), who use an unstructured language closed under negation, and Vreeswijk (1991, 1993), who uses an unstructured language containing a special sentence denoting contradiction.

Rules in a given language consist of a condition and a conclusion, which are formally a set of sentences and a sentence of the language, respectively. Since in our model all arguments are defeasible, we do not distinguish rules that give rise to strict arguments and rules that give rise to defeasible arguments.

Definition 1.

A *language* is any set, the elements of which are called the *sentences* of the language.[9] If *Subreasons* is a non-empty finite subset of a language *Language* and *Conclusion* is an element of *Language*, then

Subconditions → *Conclusion*

is a *rule* of the language *Language*.[10] The set of rules of a language *Language* is denoted as Rules(*Language*).

Fixing a language *Language*, we obtain the following formal definition of arguments in the language. Our definition of arguments in a language is related to

---

[8] This convention is similar to the conventions in the logical programming language Prolog. In fact, a simple correspondence can be given between trees of Prolog clauses and our arguments.

[9] As elements of a set *Language*, sentences are just unspecified sets. We need one formal property to avoid ambiguity: there are no sentences $Sentence_0$, ..., and $Sentence_n$ (for some natural number n), such that $Sentence_0 \in ... \in Sentence_n$.

[10] Within set theory, a rule can be defined as an ordered pair (*Subreasons*, *Conclusion*).

the definitions of Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993), but does not presuppose a set of rules. Moreover, our definition allows not only the subordination, but also the coordination of arguments. Later (definition 4) we define the rules of an argument.

Definition 2.

The set of *arguments* in the language *Language* is the smallest set such that the following hold:

1. If *Sentence* is a sentence of the language *Language*, then
   Sentence
   is an argument. The conclusion of the argument *Sentence* is *Sentence*.
2. If *Conclusion* is an element of *Language* and $Argument_1$, ..., $Argument_n$ are arguments, then
   $\{\{Argument_1, ..., Argument_n\}\} \rightarrow Conclusion$
   is an argument. The conclusion of this argument is *Conclusion*.
3. If $\{Arguments_1\} \rightarrow Conclusion$, ..., $\{Arguments_n\} \rightarrow Conclusion$ are arguments, then
   $\{Arguments_1, ..., Arguments_n\} \rightarrow Conclusion$
   is an argument. The conclusion of this argument is *Conclusion*.

The conclusion of an argument *Argument* is denoted as Conclusion(*Argument*).

The first part of the recursive definition allows statements as arguments, the second allows subordination of arguments, and the third coordination. The previously discussed argument structures are all captured by this definition. An overview is given in Table 1. The abundance of brackets { } is required to distinguish reasons and subreasons: reasons are represented as sets, and coordinated reasons as sets of sets.[11]

It may seem that there is an ambiguity between a rule and a single-step argument. However, a rule has the form $SetOfSentences \rightarrow Sentence$, where *SetOfSentences* is a set of sentences, while a single-step argument has the form $SetOfSetsOfSentences \rightarrow Sentence$, where *SetOfSetsOfSentences* is a set of sets of sentences.[12]

[11] We use sets instead of sequences, as used by Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993), since changing the order of reasons or subreasons does not change an argument.

[12] Here we need the property mentioned in note 9.

Informal argument structure	**Formal argument structure**
Sentence.	*Sentence*
Reason. So, *Conclusion.*	{{*Reason*}} → *Conclusion*
Subreason$_1$, ..., *Subreason*$_n$. So, *Conclusion.*	{{*Subreason*$_1$, ..., *Subreason*$_n$}} → *Conclusion*
Reason. So, *Conclusion*$_1$. So, *Conclusion*$_2$.	{{{{*Reason*}} → *Conclusion*$_1$}} → *Conclusion*$_2$
Reason$_1$; *Reason*$_2$. So, *Conclusion.*	{{*Reason*$_1$}, {*Reason*$_2$}} → *Conclusion*
Subreason$_{11}$, *Subreason*$_{12}$; *Subreason*$_{21}$, *Subreason*$_{22}$. So, *Conclusion.*	{{*Subreason*$_{11}$, *Subreason*$_{12}$}, {*Subreason*$_{21}$, *Subreason*$_{22}$}} → *Conclusion*
Subreason$_{11}$, ..., *Subreason*$_{1n_1}$; ...; ...; ...; *Subreason*$_{m1}$, ..., *Subreason*$_{mn_m}$. So, *Conclusion.*	{{*Subreason*$_{11}$, ..., *Subreason*$_{1n_1}$}, ..., {*Subreason*$_{m1}$, ..., *Subreason*$_{mn_m}$}} → *Conclusion*

Table 1: Overview of informal and formal argument structures

The following definitions of the premises and the rules of an argument follows the recursive structure of the definition of arguments. Vreeswijk's (1991, 1993) definitions are similar in style.

Definition 3.

If *Argument* is an argument, the set of *premises* of the argument, denoted as Premises(*Argument*), is defined recursively as follows:

1. Premises(*Sentence*) = {*Sentence*}, where *Sentence* is a sentence.
2. Premises({{*Argument*$_1$, ..., *Argument*$_n$}} → *Conclusion*) =
   Premises(*Argument*$_1$) ∪ ... ∪ Premises(*Argument*$_n$),
   where *Argument*$_1$, ..., and *Argument*$_n$ are arguments (for some natural number n), and *Conclusion* is a sentence.

3. Premises({*Arguments*$_1$, ..., *Arguments*$_n$} → *Conclusion*) =
   Premises[*Arguments*$_1$] ∪ ... ∪ Premises[*Arguments*$_n$],[13]
   where *Arguments*$_1$, ..., and *Arguments*$_n$ are sets of arguments (for some natural number n), and *Conclusion* is a sentence.

Definition 4.

If *Argument* is an argument, the set of *rules* of the argument, denoted as Rules(*Argument*), is defined recursively, as follows:

1. Rules(*Sentence*) = ∅, where *Sentence* is a sentence.
2. Rules({{*Argument*$_1$, ..., *Argument*$_n$}} → *Conclusion*) =
   {{Conclusion(*Argument*$_1$), ..., Conclusion(*Argument*$_n$)} → *Conclusion*}
   ∪ Rules(*Argument*$_1$) ∪ ... ∪ Rules(*Argument*$_n$),
   where *Argument*$_1$, ..., and *Argument*$_n$ are arguments, and *Conclusion* is a sentence.
3. Rules({*Arguments*$_1$, ..., *Arguments*$_n$} → *Conclusion*) =
   Rules[*Arguments*$_1$] ∪ ... ∪ Rules[*Arguments*$_n$],
   where *Arguments*$_1$, ..., and *Arguments*$_n$ are sets of arguments, and *Conclusion* is a sentence.

Rules are not the same as single-step arguments: the single-step argument {{*Subreason*$_1$, ..., *Subreason*$_n$}} → *Conclusion* has one rule, namely {*Subreason*$_1$, ..., *Subreason*$_n$} → *Conclusion*.

Next we define argument schemes and their instances. Argument schemes are basically arguments that can contain wildcards. An instance of an argument scheme is obtained by 'filling in' each occurrence of the wildcard *.

Argument schemes are useful to denote arguments that have a common part, such as the same final step. For instance, all arguments with an equal final step, informally denoted as

...
So, *Reason*.
So, *Conclusion*.

are represented by the argument scheme {{**Reason*}} → *Conclusion*.

In the definition of argument schemes, the wildcard * has different roles depending on its position in the argument scheme. The argument scheme **Conclusion* represents an argument with conclusion *Conclusion*. Some of the instances of **Conclusion* are *Conclusion* and {{*Reason*}} → *Conclusion*. In {{*}} → *Conclusion*, the wildcard represents any argument. Some instances are {{*Reason*}} → *Conclusion* and {{{{*Reason*$_1$}} → *Reason*$_2$}} → *Conclusion*. In {*} → *Conclusion*,

[13] For a function F and a set *Set* that is a subset of the domain of F, F[*Set*] denotes the image of *Set* under F.

the wildcard represents any (finite) set of arguments. Some instances are $\{\{Reason_1\}\} \rightarrow Conclusion$ and $\{\{Reason_1\}, \{Reason_2\}\} \rightarrow Conclusion$.

Formally, argument schemes and their instances scheme are defined as follows.

Definition 5.

The set of *argument schemes* in the language *Language* is the smallest set such that the following hold:

1. If *Sentence* is an element of *Language*, then *Sentence*, $*Sentence$, and $*$ are argument schemes.
2. If $ArgumentScheme_1$, ..., $ArgumentScheme_n$ are argument schemes, then $\{\{ArgumentScheme_1, ..., ArgumentScheme_n\}\} \rightarrow Conclusion$ and $\{*\} \rightarrow Conclusion$ are argument schemes.
3. If $\{ArgumentSchemes_1\} \rightarrow Conclusion$, ..., $\{ArgumentSchemes_n\} \rightarrow Conclusion$ are argument schemes, then $\{ArgumentSchemes_1, ..., ArgumentSchemes_n\} \rightarrow Conclusion$ is an argument scheme.

Definition 6.

The *instances* of an argument scheme *ArgumentScheme* in the language *Language*, denoted as Instances(*ArgumentScheme*), are defined recursively, as follows:[14]

1. Instances(*Sentence*) = {*Sentence*}
   Instances(*Sentence) =
   {*Argument* | *Argument* is an argument of *Language* with conclusion *Sentence*}
   Instances(*) = {*Argument* | *Argument* is an argument of *Language*}
2. Instances($\{\{ArgScheme_1, ..., ArgScheme_n\}\} \rightarrow Conclusion$) =
   Instances($ArgScheme_1$) $\cup$ ... $\cup$ Instances($ArgScheme_n$)
   Instances($\{*\} \rightarrow Conclusion$) =
   {*Arguments* $\rightarrow$ *Conclusion* | *Arguments* is a set of arguments of *Language*}
3. Instances($\{ArgSchemes_1, ..., ArgSchemes_n\} \rightarrow Conclusion$) =
   Instances[$ArgSchemes_1$] $\cup$ ... $\cup$ Instances[$ArgSchemes_n$]

Any argument is an argument scheme, whose only instance is the argument itself.

2.4 Initials and narrowings of arguments

In this section, we discuss the initials and the narrowings of an argument. They are purely determined by the structure of the argument.

Arguments can have other arguments as initial parts. For instance, the argument

---

[14] This definition is recursive in the structure of arguments, just as the definitions of premises, rules and argument schemes. For brevity, we do not explicitly state that *Sentence* is a sentence, that $Argument_i$ is an argument (for i = 1, ..., n) etc.

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

has the argument

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

as an initial part. Formally the initials of an argument are defined as follows.

Definition 7.

If *Argument* is an argument, the set of *initials* of the argument, denoted as Initials(*Argument*), is defined recursively as follows

1. Initials(*Sentence*) = ∅
2. Initials({{*Argument*$_1$, ..., *Argument*$_n$}} → *Conclusion*) = Initials(*Argument*$_1$) ∪ ... ∪ Initials(*Argument*$_n$) ∪ {*Argument*$_1$, ..., *Argument*$_n$}
3. Initials({*Arguments*$_1$, ..., *Arguments*$_n$} → *Conclusion*) = Initials({*Arguments*$_1$} → *Conclusion*) ∪ ... ∪ Initials({*Arguments*$_n$} → *Conclusion*)

The initials of an argument are also arguments. The definition shows that an argument is not an initial of itself and that all arguments, except for statements, have initials.

If the conclusion of an argument is supported by a coordinate argument with separate reasons, one or more of the reasons can be removed from the argument. For instance, if the reason 'The sun is shining' is removed from the argument

The sun is shining; The sky is blue.
So, it is a beautiful day.

we obtain the argument

The sky is blue.
So, it is a beautiful day.

The latter argument is called a narrowing of the former. Only if one allows the coordination of arguments, it is possible to define the narrowings of an argument. Formally, the narrowings of an argument are defined as follows.

Definition 8.

If *Argument* is an argument, the set of *narrowings* of the argument, denoted as Narrowings(*Argument*), is defined recursively as follows:

1. Narrowings(*Sentence*) = ∅
2. Narrowings({{*Argument*$_1$, ..., *Argument*$_n$}} → *Conclusion*) =
   {{{*Narrowing*$_1$, ..., *Narrowing*$_n$}} → *Conclusion* |
   Narrowing$_i$ ∈ Narrowings(*Argument*$_i$) for all i = 1, ..., n}
3. Narrowings({*Arguments*$_1$, ..., *Arguments*$_n$} → *Conclusion*) =
   {*W* → *Conclusion* | ∅ ⊂ *W* ⊂ {*Arguments*$_1$, ..., *Arguments*$_n$}}[15]

If *Argument*$_1$ is a narrowing of *Argument*$_2$, then *Argument*$_2$ is a *broadening* of *Argument*$_1$.

The definition shows that in a narrowing of an argument the final conclusion is supported by less reasons than in the argument itself (part 3 of the definition). In a narrowing of an argument, also the intermediate conclusions can be supported by less reasons (part 2 of the definition).

The narrowings of arguments are also arguments. If follows from the definition that arguments are not narrowings of themselves, and that not all arguments have narrowings. The conclusion of a narrowing of an argument is equal to the conclusion of the argument. As a result, no narrowing of an argument is at the same time an initial of the argument.

3 Defeat and defeaters

In the previous section, we saw that arguments are in form comparable to proofs. However, there is a major difference between arguments and proofs. While proofs justify their conclusions under all circumstances, arguments do not: arguments can be defeated.

In this section, we deal with the defeat of arguments. We distinguish several types of defeat and corresponding defeaters. These indicate which arguments can defeat which other arguments (sections 3.1 to 3.5). Then we discuss the role of defeater schemes (section 3.6). This leads to the formal definition of defeaters and defeater schemes (section 3.7).

3.1 Undercutting defeat

The first type of defeat that we discuss is defeat by an undercutter.[16] As an example, we consider the following (single-step) argument:

The object looks red.
So, the object is red.

---

[15] V ⊂ W means that V is a proper subset of W.

[16] Pollock (1987-1995) has argued for the distinction between defeat by an undercutter and by a rebutter (discussed in section 3.2).

In principle this argument supports its conclusion, but suppose that we also have the following argument (a statement):

The object is illuminated by a red light.

Taking both arguments into account, the argument that the object is red does no longer justify its conclusion. Because the object is illuminated by a red light, the fact that it looks red is no longer a reason for the conclusion that the object is red. (Of course the object can still be red, but we cannot justify this by the fact that it looks red.) We say that the fact that the object is illuminated by a red light *undercuts* the argument that the object is red.[17]

In our formal model, this fact is represented as follows:

Illuminated_by_a_red_light [{{Looks_red}} → Is_red]

This is an example of a *defeater*, the formal definition of which follows later.[18] Informally, the defeater represents that if the argument on the left, Illuminated_by_a_red_light, is undefeated, it defeats the argument on the right {{Looks_red}} → Is_red.[19] To emphasize that the latter argument becomes defeated, it is put between square brackets [ ].

3.2 Rebutting defeat

The second type of defeat is rebuttal.[20] For instance, if John likes French fries, but is on a low calorie diet, we have the following two arguments:

John likes French fries.
So, he orders French fries.

and

---

[17] The example has been used at several occasions by Pollock as an illustration of undercutting defeat (e.g., Pollock, 1986, p. 39ff.; 1994).

[18] In our terminology, a defeater is not itself an argument or a reason that challenges another argument (as for instance Pollock uses the term), but a relation between challenging and challenged arguments.

[19] Note that the arguments Illuminated_by_a_red_light and {{Looks_red}} → Is_red do not have inconsistent conclusions. The example shows an important choice underlying the CumulA model: the defeat of arguments is in CumulA not *inconsistency-triggered*, but *counterargument-triggered* (see note 5). Not inconsistency, but counterargument (represented by defeaters) is the primitive notion in CumulA. We come back to this distinction in chapter 6, section 4.

[20] See note 16.

John is on a low calorie diet.
So, he does not order French fries.

Assuming that people who are on a diet try to suppress their eating impulses, John probably does not order fries, since the latter argument would be more important. In this case, the former argument is defeated by the latter. Formally, this would be represented by the following defeater:

{{On_low_calorie_diet}} → Not_order_fries [{{Likes_fries}} → Order_fries]

The argument on the left, {{On_low_calorie_diet}} → Not_order_fries, defeats the argument in square brackets on the right, {{Likes_fries}} → Order_fries. If, as in this example, an argument defeats an argument with opposite conclusion, we speak of *rebutting* defeat.

3.3 Defeat by sequential weakening

The third type of defeat is defeat by sequential weakening. An example of this is the following argument, based on the well-known sorites paradox:[21]

This body of grains of sand is a heap.
So, this body of grains of sand minus 1 grain is a heap.
So, this body of grains of sand minus 2 grains is a heap.
...
So, this body of grains of sand minus *n* grains is a heap.

Each single step of the argument is correct, but clearly the argument cannot be pursued indefinitely, since in the end there is no grain of sand left. For *n* large enough, the argument above does clearly not justify its conclusion and should be defeated. The important point here is that it is impossible to choose a single step that makes the argument defeated. Only because the step is repeated too often, the argument is weakened below the limit of acceptability, and is defeated.

Since argument steps normally can be chained, we need a way to represent the fact that certain sequences of steps can lead to the defeat of an argument. A defeater representing the situation of our example has the following form:

[Body_of_sand_is_heap → Body_of_sand_minus_1_grain_is_heap
→ Body_of_sand_minus_2_grains_is_heap
→ ...
→ Body_of_sand_minus_*n*_grains_is_heap]

For convenience, we have left out the brackets { }.

21 Read (1995, p. 173ff.) discusses philosophical issues related to the sorites paradox.

In the example, there is an argument that is clearly defeated because it contains an unacceptable sequence of steps, but that does not contain one single argument step that is to blame. In such a case, we speak of defeat by *sequential weakening* of the argument.[22]

3.4 Defeat by parallel strengthening

The fourth type of defeat is defeat by parallel strengthening. Assume that John has committed an offense, but is a minor first offender. As a result, the judge might consider the following argument:

John is a minor first offender.
So, John should not be punished.

If for instance John has robbed Alex, the judge might consider this an argument that rebuts the following argument with opposite conclusion:

John has robbed Alex.
So, John should be punished.

In the case of rebuttal the judge decided not to punish John. The judge might decide analogously if John had injured Alex in a fight.

However, if John has both robbed Alex and injured him in a fight, the judge might decide differently. Since there are now two reasons for punishing John, coordination of the arguments gives us the following composite argument:

John has robbed Alex; John injured Alex in a fight.
So, John should be punished.

This argument might defeat the argument that John should not be punished. In that case, the argument that John should be punished defeats another argument, while its narrowings do not. A defeater representing this is the following:

{{Robbed}, {Injured}} → Punished [{{Minor_first_offender}} → Not_punished]

The argument {{Robbed}, {Injured}} → Punished defeats the argument {{Minor_first_offender}} → Not_punished.[23] In this example the defeat of the argument not to punish John can be explained by the *parallel strengthening* of the argument to punish him. We speak of defeat by parallel strengthening if an

22 The term is taken from Verheij (1995c).
23 In Reason-Based Logic, this example would involve the weighing of reasons (chapter 2, sections 1.3 and 3.3). Cf. Verheij (1994).

argument that has narrowings defeats another argument, while the narrowings themselves do not.[24]

3.5 Collective and indeterministic defeat

All examples of defeaters that we have seen consisted of a single challenging and a single challenged argument. Such defeaters are called simple. However, there are cases in which groups of arguments must be considered. We discuss two types of situations in which this is the case, namely *collective defeat* and *indeterministic defeat*. It turns out that in order to represent these types of defeat, we need compound defeaters, that consist of groups of challenging and challenged arguments.

Collective and indeterministic defeat occur if there is a number of arguments that can clearly not all justify their conclusions, for instance because their conclusions cannot all hold, but neither of which is clearly defeated by any of the others. We give an example.

It can be the case that an employer wants to hire two persons if they are qualified. If John is qualified, the employer can make the following argument:

John is qualified. So, John is hired.

On its own, this argument can be undefeated, but now assume that not only John, but also Alex and Mary are qualified for the job. As a result, the employer can also make the following two arguments:

Alex is qualified. So, Alex is hired.
Mary is qualified. So, Mary is hired.

Since the employer only wants to hire two persons, the three arguments cannot all be undefeated.

If there is no additional information to resolve this conflict of arguments, two approaches can be distinguished that nevertheless 'magically' resolve the conflict: collective and indeterministic defeat.

In the first approach to dealing with the unresolved conflict of arguments, collective defeat,[25] all arguments are considered defeated. We speak of collective defeat if a group of arguments is defeated as a whole, while the arguments in the

[24] The example given here is also an example of rebutting defeat, showing that the discussed types of defeat can overlap.

[25] The term 'collective defeat' stems from Pollock (1987). Our collective defeat generalizes his. Pollock's collective defeat is a general principle to preserve consistency. It makes groups of otherwise undefeated, but conflicting arguments defeated. Our collective defeat is optional, depending on the compound defeaters of a particular argumentation theory, and can occur for any group of arguments, not only conflicting.

group would on their own not be defeated. A defeater representing the situation in our example could have the following form:

[{{John_is_qualified}} → John_is_hired,
{{Alex_is_qualified}} → Alex_is_hired,
{{Mary_is_qualified}} → Mary_is_hired]

In this defeater, all arguments are inside the square brackets indicating that they are defeated as a group. We say that this defeater is *right-compound*, since it has more than one challenged argument.

A defeater such as the one above represents that the arguments inside the square brackets are defeated as a group, and not simply that they are all three defeated. The latter would be represented by the following three simple defeaters:

[{{John_is_qualified}} → John_is_hired]
[{{Alex_is_qualified}} → Alex_is_hired]
[{{Mary_is_qualified}} → Mary_is_hired]

The difference with the compound defeater above is that the compound defeater only represents that the group of three should be defeated if otherwise neither of the arguments in the group would be defeated. If the argument that Mary is hired for the job is defeated *for another reason* (i.e., because of another defeater), for instance, that she prefers a job somewhere else, the compound defeater above does not anymore imply the defeat of the argument that John is hired for the job. Only if all three arguments would otherwise be undefeated, the compound defeater results in their defeat as a group. The three simple defeaters would not have the same effect: they represent that the arguments are defeated anyway.

The second approach to dealing with the unresolved conflict of arguments is indeterministic defeat. In this approach, the conflict is resolved by considering one of the arguments in the conflict defeated. Since there are several choices that can be made, neither of which is better than the others, the conflict is 'indeterministically' solved: each choice of a defeated argument is allowed. In the example, there are three solutions, represented by the following defeaters:

{{John_is_qualified}} → John_is_hired, {{Alex_is_qualified}} → Alex_is_hired
[{{Mary_is_qualified}} → Mary_is_hired]
{{Alex_is_qualified}} → Alex_is_hired, {{John_is_qualified}} → John_is_hired
[{{Mary_is_qualified}} → Mary_is_hired]
{{Mary_is_qualified}} → Mary_is_hired, {{John_is_qualified}} → John_is_hired
[{{Alex_is_qualified}} → Alex_is_hired]

Each defeater represents that two of the arguments challenge the third, and can result in its defeat if they are both undefeated. We say that this defeater is *left-compound*, since it has more than one challenging arguments.

3.6 Defeater schemes

We have encountered several examples of defeaters. They all contained representations of full arguments. As we will see, this is not always convenient. As an example, we reconsider the argument that an object is red because it looks red. This argument was defeated by the statement that the object is illuminated by a red light. We had the following defeater:

Illuminated_by_a_red_light [{{Looks_red}} → Is_red]

But it can of course also be the case that the fact 'The object is illuminated by a red light' is not merely put forward as a statement, but is itself supported by some non-trivial argument, for instance as follows:

Ralph says that the object is illuminated by a red light.
So, the object is illuminated by a red light.

If this argument is not defeated, it defeats the argument that the object is red, just like the statement 'The object is illuminated by a red light' did. It does not matter how the conclusion that the object is illuminated by a red light is justified. By whatever argument that conclusion is justified, it defeats the argument that the object looks red.

Similarly, it can be the case that the argument step that the object is red because it looks red is itself part of a larger argument, for instance as follows:

The object reflects light of a particular wave length.
So, the object looks red.
So, the object is red.

This argument is defeated too if the conclusion that the object is illuminated by a red light is justified. (It should be noted that this does not imply that the argument

The object reflects light of a particular wave length.
So, the object looks red.

is defeated. The conclusion that the object looks red is still justified.)

This leads to the notion of *defeater schemes*. We want to represent that any argument that justifies the conclusion that the object is illuminated by a red light, defeats any argument that ends with the argument step that the object is red because it looks red. A defeater scheme representing this looks as follows:

*Illuminated_by_a_red_light [{{*Looks_red}} → Is_red]

Instead of arguments, a defeater scheme contains argument schemes, such as *Illuminated_by_a_red_light and {{*Looks_red}} → Is_red. The defeater above will have the effect that any argument that is an instance of *Illuminated_by_a_red_light challenges any instance of {{*Looks_red}} → Is_red.

In our example, this is just as required, since the argument scheme {{*Looks_red}} → Is_red has both

{{Looks_red}} → Is_red

and

{{{{Reflect_light_of_particular_wave_length}} → Looks_red}} → Is_red

as instances.

3.7 Definition of defeaters and argumentation theories

Having finished the description of different types of defeat, we come to the formal definition of defeaters. We have seen several examples, all captured by the following definition.

Definition 9.

If $ChallengingArgument_1$, ..., $ChallengingArgument_n$, $ChallengedArgument_1$, ..., $ChallengedArgument_m$ are arguments, then

$ChallengingArgument_1$, ..., $ChallengingArgument_n$
[$ChallengedArgument_1$, ..., $ChallengedArgument_m$]

is a *defeater*. The arguments $ChallengingArgument_1$, ..., $ChallengingArgument_n$ are the *challenging arguments* of the defeater, the arguments $ChallengedArgument_1$, ..., $ChallengedArgument_m$ the *challenged arguments* of the defeater. A defeater with at most one challenging and at most one challenged argument is *simple*, otherwise *compound*. A defeater that has more than one challenging argument is *left-compound*, a defeater with more than one challenged argument *right-compound*.

Intuitively, the defeater $ChallengingArgument_1$, ..., $ChallengingArgument_n$ [$ChallengedArgument_1$, ..., $ChallengedArgument_m$] represents the fact that the arguments $ChallengingArgument_1$, ..., and $ChallengingArgument_n$ defeat the arguments $ChallengedArgument_1$, ..., $ChallengedArgument_m$, if they are themselves not defeated.

Our defeaters are related to Dung's (1993, 1995) attacks. However, our defeaters take the structure of arguments into account, and do not consist of a single challenging and a single challenged argument, but of a group of challenging and a group of challenged arguments.

We already discussed the need for defeater schemes. They contain argument schemes instead of arguments. Defeater schemes and their instances are formally defined as follows.

Definition 10.

If *ChallengingArgScheme*$_1$, ..., *ChallengingArgScheme*$_n$, *ChallengedArgScheme*$_1$, ..., *ChallengedArgScheme*$_m$ are argument schemes, then

ChallengingArgScheme$_1$, ..., *ChallengingArgScheme*$_n$ [*ChallengedArgScheme*$_1$, ..., *ChallengedArgScheme*$_m$]

is a *defeater scheme*. If each argument scheme in the defeater scheme is replaced by one of its instances, the resulting defeater is an *instance* of the defeater scheme.

Just as arguments are a special kind of argument schemes, defeaters are a special kind of defeater schemes, with as only instance the defeater itself.

We have described several types of defeat. In Table 2, we give an overview of these types of defeat and their corresponding defeater schemes.

Type of defeat	**Corresponding defeater scheme(s)**
Undercutting defeat	$*Conclusion_1$ [$*Reason \rightarrow Conclusion_2$]
Rebutting defeat	$*Pro \rightarrow Conclusion$ [$*Con \rightarrow Not\_conclusion$]
Defeat by sequential weakening	[$*Sentence_1 \rightarrow ... \rightarrow Sentence_n$]
Defeat by parallel strengthening	{{$*Reason_1$}, ..., {$*Reason_n$}} $\rightarrow Conclusion_1$ [$*Conclusion_2$]
Collective defeat	[$*Conclusion_1$, ..., $*Conclusion_n$]
Indeterministic defeat	$*Conclusion_1$ [$*Conclusion_2$, ..., $*Conclusion_n$], ..., $*Conclusion_n$ [$*Conclusion_1$, ..., $*Conclusion_{n-1}$],

Table 2: Types of defeat and their corresponding defeater schemes

The types of defeat in this table are not disjoint, in the sense that there can be defeaters that are instances of defeater schemes of different types.

Argumentation depends on which arguments can support conclusions and on the situations in which arguments defeat other arguments. This is specified by an *argumentation theory*. A natural way to specify the arguments is by the rules from which they are constructed. A natural way to specify defeat situations is by defeater schemes. This gives us the following definition.

Definition 11.

An *argumentation theory* is a triple (*Language*, *Rules*, *DefeaterSchemes*), such that *Language* is a language, *Rules* is a set of rules of the language *Language*, and *DefeaterSchemes* is a set of defeater schemes of the language. Any argument that has only rules in *Rules* is an *argument* of the argumentation theory. Any instance of a defeater scheme in *DefeaterSchemes* is a *defeater* of the argumentation theory.

Our argumentation theories correspond to Lin and Shoham's argument structures (Lin and Shoham, 1989; Lin, 1993), Vreeswijk's (1991, 1993) abstract argumentation systems, and Dung's (1993, 1995) argumentation frameworks. Lin and Shoham and Vreeswijk include a set of premises. In CumulA, however, an argumentation theory does not specify premises, since in CumulA's lines of argumentation the premises can change (see section 5). Dung's definitions do not specify a set of premises either, but for the reason that they fully abstract from the structure of arguments.

4 Stages of the argumentation process

In the previous two sections, we discussed arguments and defeaters. They play a central role in argumentation: arguments represent how conclusions can be supported and defeaters represent which arguments can defeat other arguments. In this section, we discuss how defeaters determine the status of the arguments taken into account, i.e. which of them are defeated and which are undefeated. In the next subsection we treat how the status of an argument relates to the status of its initials and narrowings (section 4.1). Then we describe some notions that characterize the effects of a defeater (section 4.2). Thereafter we characterize the status of arguments in an argumentation stage (section 4.3). In section 4.4, the stages of an argumentation theory are formally defined.

4.1 Initials, narrowings and defeat

In this section, we encounter three general requirements that must hold for any defeat status assignment of the arguments that are taken into account at a stage of the argumentation process.

Every argument that is taken into account has one of two statuses: it can be either undefeated or defeated. By definition, an undefeated argument justifies its

conclusion, while a defeated argument does not. So, the first requirement is simply that no argument can be defeated and undefeated at the same time. Obviously, an argument cannot both justify and not justify its conclusion.

The second requirement relates the statuses of an argument and its initials: if an initial of an argument is defeated, the argument is itself defeated. For instance, if the argument

The object looks red.
So, the object is red.

is defeated and does not justify the conclusion that the object is red, the argument

The object looks red.
So, the object is red.
So, the object attracts the attention.

is also defeated and cannot justify the conclusion that the object attracts the attention. (On the other hand, it is possible that the latter argument is defeated, while the former is not.) Generally, an argument can not justify its conclusion if an intermediate conclusion is not justified. In other words, an argument never withstands defeat better than its initials.

The third requirement relates the statuses of an argument and its narrowings: if a narrowing of an argument is undefeated, the argument is itself undefeated. For instance, if the argument

The sun is shining.
So, it is a beautiful day.

is undefeated and justifies its conclusion that it is a beautiful day, the argument

The sun is shining; The sky is blue.
So, it is a beautiful day.

cannot be defeated and not justify that same conclusion. Intuitively, an argument does not withstand defeat worse than an argument containing less reasons.[26] Only one of the narrowings of an argument needs to be undefeated in order to make it undefeated. For instance, the argument

The sky is blue.
So, it is a beautiful day.

[26] Pollock (1991) has argued against this so-called accrual of reasons. In chapter 6, section 2, we give our reply.

can be defeated (by some other argument), while the two arguments above are undefeated. Adding a reason can never make an argument defeated, but sometimes can make an argument undefeated, as in a case of defeat by parallel strengthening.

Summarizing we have the following three requirements for any defeat status assignment:

1. Each argument is either undefeated or defeated.
2. If an initial of an argument is defeated, the argument is defeated.
3. If a narrowing of an argument is undefeated, the argument is undefeated.

4.2 Relevant, triggered, respected and inactive defeaters

Defeaters play a central role in the determination of the defeat status of arguments. By default, an argument is undefeated, but defeaters can change this default status. Recall that defeaters indicate when undefeated arguments can defeat other arguments. We discuss four notions that are important for the effects of defeaters: a defeater can be *relevant*, *triggered*, *respected* and *inactive*.

As an example, we take the defeater

Ralph's_testimony → Illuminated_by_a_red_light [Looks_red → Is_red]

A defeater only can have effects if all arguments in it have been taken into account. If only the argument Looks_red → Is_red has been taken into account, the defeater above has no effect. Only if the argument Ralph's_testimony → Illuminated_by_a_red_light is also taken into account, can the argument Looks_red → Is_red be challenged. If all arguments in a defeater have been taken into account, the defeater is *relevant*.

A relevant defeater can only lead to the defeat of its challenged arguments if the challenging arguments are undefeated. Returning to our example, it can turn out that Ralph is lying, with the result that the argument Ralph's_testimony → Illuminated_by_a_red_light does not justify its conclusion, and is defeated (on the basis of some other defeater). Even though the argument that the object is illuminated by a red light is taken into account, it does not challenge the argument that the object is red, since it is itself defeated. If all challenging arguments of a relevant defeater are undefeated, the defeater is *triggered*.

Normally, if a relevant defeater is triggered, the challenged arguments are defeated. In our example, if the argument Ralph's_testimony → Illuminated_by_a_red_light is undefeated, the argument Looks_red → Is_red is defeated. In general, if all challenged arguments of a triggered defeater are defeated, the defeater is *respected*.

There is one situation in which a defeater of which the challenging arguments are undefeated do not lead to the defeat of its challenged arguments. This can only happen in a case of collective defeat (see section 3.5), represented by a right-compound defeater, when the challenged arguments are not defeated as a group,

because some of them (but not all) are challenged arguments of another respected defeater.

As an example, we take the following two defeaters:

[John_is_qualified → John_is_hired, Mary_is_qualified → Mary_is_hired]
Mary_prefers_another_job [Mary_is_qualified → Mary_is_hired]

Here Mary_prefers_another_job represents that Mary prefers another job than the one for which both John and Mary are qualified. If now the two arguments

John_is_qualified → John_is_hired
Mary_is_qualified → Mary_is_hired

are taken into account, only one of the defeaters is relevant (the first), and should result in the defeat of both arguments. But if also the statement

Mary_prefers_another_job

is taken into account, the situation changes. Both defeaters are relevant. Since the argument Mary_prefers_another_job is not even challenged in one of the defeaters, it is undefeated and defeats the argument Mary_is_qualified → Mary_is_hired. But now the other defeater, that represents collective defeat, should not lead to the defeat of John_is_qualified → John_is_hired, since one of its challenged arguments is already defeated by another defeater. In this situation, we say that the defeater is *inactive*, otherwise *active*. Only active defeaters can lead to the defeat of their challenged arguments.

4.3 Stages and defeat

The stages of the argumentation process are characterized by the arguments that have been taken into account, and by the status the arguments have. The status of the arguments taken into account is determined by the defeaters of the argumentation theory. In this section, we discuss how.

An argument can be defeated in two ways: directly and indirectly. An argument is *directly defeated* if it is a challenged argument of a triggered active defeater. As an example, we consider the red light example again. We have the simple defeater

Illuminated_by_a_red_light [Looks_red → Is_red]

Assume that two arguments have been taken into account:

Illuminated_by_a_red_light
Looks_red → Is_red

Clearly, the defeater is relevant. The argument Illuminated_by_a_red_light cannot be defeated, since there is no defeater in which it is challenged. As a result, the defeater is triggered, and also active, since it is not left-compound. As a result, the argument Looks_red → Is_red is directly defeated by the argument Illuminated_by_a_red_light.

An argument is *indirectly defeated* if it has a defeated initial or broadening. Indirect defeat corresponds to the requirements on the statuses of arguments and their initials and narrowings (section 4.1). For instance, if in the example above the argument

Looks_red → Is_red → Attracts_attention

were also taken into account, it would be defeated, because its initial Looks_red → Is_red would have been defeated.

An argument can be both directly and indirectly defeated. For instance, if the argument

John_is_color_blind

were taken into account, and the theory contained the defeater scheme

John_is_color_blind [*Is_red → Attracts_attention],

the argument Looks_red → Is_red → Attracts_attention would be both directly defeated, by the argument John_is_color_blind, and indirectly, by the (direct) defeat of its initial Looks_red → Is_red.

We have now discussed all ingredients of our definition of an argumentation stage. A stage is characterized by the arguments that are taken into account and by their statuses. As a result, a stage is defined as a defeat status assignment, that must obey the requirements of section 4.1. Furthermore, which arguments have the status undefeated and which defeated is determined by the defeaters, as follows:

An argument has the status 'defeated' if and only if the argument is directly or indirectly defeated.

4.4 Definition of stages

We have seen that the status of initials and narrowings of an argument can have an effect on the status of the argument itself. The *range* of a set includes all arguments that can have such effects for the arguments of the set. Formally, the range of a set of arguments is defined as follows.

Definition 12.

The *range* of a set of arguments *Arguments*, denoted as Range(*Arguments*), is the smallest set of arguments *Range*, such that the following hold:

1. *Arguments* is a subset of *Range*.
2. Any initial of an argument in *Range* is an element of *Range*.
3. Any narrowing of an argument in *Range* is an element of *Range*.

A set of arguments that is equal to its range is a *range* of arguments.

We have discussed three requirements that are the result of the relations of the status of an argument and the statuses of its initials and narrowings. These requirements lead to the following definition of a *defeat status assignment*.

Definition 13.

A *defeat status assignment* of a range of arguments *Range* has the form

UndefeatedArguments (*DefeatedArguments*),

such that the following hold:

1. The arguments in *Range* are precisely the arguments in *UndefeatedArguments* and *DefeatedArguments*, but no argument is both in *UndefeatedArguments* and in *DefeatedArguments*.
2. No initial of an argument in *UndefeatedArguments* is an element of *DefeatedArguments*.
3. No narrowing of an argument in *DefeatedArguments* is an element of *UndefeatedArguments*.

The set *Range*, equal to the union of *UndefeatedArguments* and *DefeatedArguments*, is the *range* of the defeat status assignment.

Notation: A defeat status assignment of a finite range of arguments will often be denoted as

$$UndefeatedArgument_1, ..., UndefeatedArgument_n$$
$$(DefeatedArgument_1, ..., DefeatedArgument_m)$$

Our defeat status assignments are formally related to Pollock's (1994, 1995) partial status assignments, but have a different use. Pollock uses status assignments to be able to deal with certain problem cases. We use status assignments since they enable the definition of argumentation stages.

The second requirement in the definition of defeat status assignments is well-known and has a counterpart (in different forms) in many argumentation models that take the subordination of arguments into account, such as the models of Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993). The third requirement is, as far as we know, new in CumulA since in other models the coordination of arguments is not taken into account. It represents how the coordination of arguments is related to defeat.

Next we define when a defeater is relevant, triggered, respected and (in)active.

Definition 14.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *relevant* for a defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if all arguments in *ChallengingArguments* and *ChallengedArguments* are in the range of the defeat status assignment.

Definition 15.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *triggered* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if it is relevant and *ChallengingArguments* is a subset of the range of *UndefeatedArguments*.

Definition 16.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *respected* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if *ChallengingArguments* is a subset of the range of *UndefeatedArguments* and *ChallengedArguments* is a subset of the range of *DefeatedArguments*.

Definition 17.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *inactive* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if it is relevant and there is a respected defeater *ChallengingArguments'* [*ChallengedArguments'*], such that some, but not all, arguments in *ChallengedArguments* are an element of, or have an initial or broadening in *ChallengedArguments'*. A relevant defeater is *active* if it is not inactive.

As immediate consequences of these definitions, triggered defeaters are always relevant, and respected defeaters are always triggered (and therefore relevant).

The following definition captures the direct and indirect defeat of arguments.

Definition 18.

The argument *Argument* is *defeated* by the arguments *ChallengingArguments* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if there is a triggered active defeater *ChallengingArguments* [*ChallengedArguments*], such that

1. *ChallengedArguments* contains *Argument*, or
2. *ChallengedArguments* contains an initial or broadening of *Argument*.

In the first case, the argument *Argument* is *directly defeated* by the arguments *ChallengingArguments*; in the second case, *indirectly*.

We finally have arrived at the formal definition of argumentation stages.

Definition 19.

An *argumentation stage* of an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*) is a defeat status assignment of a range of arguments in the language *Language* with rules in *Rules*,

UndefeatedArguments (*DefeatedArguments*),

such that the following holds:

Argument is an element of *DefeatedArguments* if and only if *Argument* is defeated by arguments that are elements of *UndefeatedArguments*.

The premises and conclusions of the arguments in the range of the argumentation stage are the *premises* and the *conclusions* of the argumentation stage, respectively. The conclusions of arguments in the range of the argumentation stage that are not an initial of another argument in the range, are the *final conclusions* of the argumentation stage. The conclusions of the arguments in *UndefeatedArguments* are the *justified conclusions* of the argumentation stage; the conclusions of arguments in *DefeatedArguments* the *unjustified conclusions*.

The constraint says that the arguments in *DefeatedArguments* are exactly the arguments that are (directly or indirectly) defeated. It turns out that a given range of arguments can correspond to zero, one or several argumentation stages of a theory. Section 6 contains examples.

Our stages are similar to Vreeswijk's (1991, 1993) argument structures. On a formal level, the definitions differ since the approaches to defeat in Vreeswijk's model and in CumulA are different (see chapter 6, section 4). Moreover, the intuitions behind Vreeswijk's arguments structures and CumulA's argumentation are different: Vreeswijk's argumentation structures represent the arguments that are currently undefeated, while CumulA's stages represent both the currently undefeated arguments and the currently defeated arguments. Verheij (1995b, c) argues that the latter is more general and closer to the idea of gradually taking arguments into account.

Verheij (1996a) investigates the relations of CumulA's stages (in a restricted form) and Dung's (1993, 1995) admissible sets of arguments. As Verheij (1996a) shows, there are close relationships on the formal level. However, Dung's admissible sets are seemingly not meant to model stages of the argumentation process. Verheij (1996a) gives examples and formal relations that show that the stages approach generalizes the admissible sets approach, and models the intuition of gradually taking arguments into account.

5 Lines of argumentation and argumentation diagrams

We consider argumentation as a process, in which arguments are taken into account, and are assigned a defeat status. Now that we have described the stages of

this process, we will discuss lines of argumentation, that are intuitively series of consecutive argumentation stages.

We treat the construction of arguments in a line of argumentation and the change of status of arguments in a line of argumentation. We finish with the formal definition of lines of argumentation and argumentation diagrams.

5.1 Construction of arguments

In a line of argumentation, arguments are gradually constructed. Since we consider a line of argumentation as a sequence of argumentation stages, the gradual construction of arguments means that the range of the stages in a line of argumentation gradually changes. We distinguish six elementary ways to construct new arguments from the arguments taken into account at some stage, leading to a new stage. We also mention how these constructions affect the premises and conclusions of the stage.

First, at any stage in a line of argumentation a *new statement* can be introduced. Moreover, a line of argumentation can start with a statement. For instance, the initial statement might be:

It is raining.

As mentioned earlier (e.g., in section 2 on arguments), we treat statements as arguments with trivial structure. At this stage of the line of argumentation, where only the statement 'It is raining' is taken into account, we have one premise and one conclusion that coincide, namely 'It is raining'. In general, if at some stage a new statement is introduced, at the new stage a (coinciding) premise and conclusion are added to those of the original stage. [27]

Second, a *forward step* can be added to an argument taken into account. This means that the conclusion of the argument is used to support a new conclusion. For instance, the statement that it is raining can be used to support whether to put on a raincoat or not. We obtain the following single-step argument:

It is raining.
So, it is wise to put on a raincoat.

If a forward step is added to an argument, the premises do not change, but a new conclusion is introduced. In the example, the new conclusion is 'It is wise to put on a raincoat'.

Third, a *backward step* can be added to an argument. This means that the premise of the argument is supported by a new premise. For instance, if I am in a room that has no windows, I might not take the statement that it is raining for

[27] It can of course be the case that such a premise or conclusion is not new because it was already a premise or conclusion of *another* argument taken into account.

granted, and look for support of the conclusion that it is raining. For instance, the following single-step argument can support that conclusion:

A colleague is completely soaked and tells that it is raining.
So, it is raining.

If a backward step is added to an argument, a premise is replaced by one or more new premises, while the conclusions remain the same. In the example, 'It is raining' is no longer a premise, and is replaced by the premise 'A colleague is completely soaked and tells that it is raining'.

Fourth, a *broadening step* can be added to an argument. This means that the conclusion of a (non-trivial) argument is supported by an additional reason. For instance, it might be the case that the conclusion that it is raining gets additional support by the weather report on the radio. In that case, the previous argument can be broadened to the following argument:

A colleague is completely soaked and tells that it is raining; The weather-report on the radio says that is raining.
So, it is raining.

If a broadening step is added to an argument, the conclusions of the original stage remain the same, while new premises are introduced. In the example, 'The weather-report on the radio says that is raining' is a new premise.

Fifth, two arguments can be combined by *subordination* if one of the arguments taken into account has a premise that is the conclusion of the other. In this way, an argument taken into account can be used to support the premise of another argument. For instance, the argument

A colleague is completely soaked and tells that it is raining.
So, it is raining.

can be subordinated to the argument

It is raining.
So, it is wise to put on a raincoat.

This results in the argument

A colleague is completely soaked and tells that it is raining.
So, it is raining.
So, it is wise to put on a raincoat.

In a case of subordination, a premise and a conclusion of the original stage can be dropped at the new stage.[28] In the example, the premise 'It is raining' is dropped.

Sixth, two arguments can be combined by *coordination* if they have the same conclusion. For instance, the arguments

A colleague is completely soaked and tells that it is raining.
So, it is raining.

and

The weather-report on the radio says that is raining.
So, it is raining.

can be coordinated, resulting in the argument

A colleague is completely soaked and tells that it is raining; The weather-report on the radio says that is raining.
So, it is raining.

In a case of coordination, the premises and conclusions of the original stage remain the same at the new stage.

Summarizing, we distinguished six types of argument construction:

1. Introducing a new statement
2. Adding a forward step
3. Adding a backward step
4. Adding a broadening step
5. Subordinating one argument to another
6. Coordinating two arguments

Each of these types has an inverse, that can be considered as a type of argument deconstruction. For instance, the inverse of the introduction of a statement is the withdrawal of a statement. However, we focus on argument construction.[29]

5.2 Change of status

Argumentation stages are characterized by the arguments taken into account and by their status. It is characteristic for argumentation with defeasible arguments that the status of arguments can change in a line of argumentation.

---

[28] It can of course be the case that such a premise or conclusion is not dropped because it is still a premise or conclusion of *another* argument taken into account (cf. note 27).

[29] Technically, as we will see, we will define lines of argumentation in terms of argument construction. Argument deconstruction can be considered as *backtracking* in a line of argumentation.

A basic example of the change of status is reinstatement. In a case of reinstatement, an argument is undefeated at some stage, defeated at a second, later stage, and again undefeated at a third, again later stage. For instance, the argument

The_object_looks_red → The_object_is_red

can first be undefeated, then defeated by the statement

Ralph_says_the_object_is illuminated_by_red_light,

and again undefeated by the statement

Ralph_is_a_liar.

Reinstatement depends on the order in which arguments are taken into account. For instance, if in some line of argumentation the statement that Ralph is a liar was taken into account first, the argument that the object is red would not become defeated.

If we abbreviate the three arguments above as α, β and γ, respectively, all lines of argumentation, corresponding to the six orders in which the three arguments can be taken into account, can be summarized in a so-called argumentation diagram (Figure 1). The nodes in the diagram correspond to argumentation stages. The 0 corresponds to the stage with empty range, at which no arguments have been taken into account. If an argument is defeated in a stage, it is denoted in brackets. The arrows indicate the transition from one stage to the next in a line of argumentation.

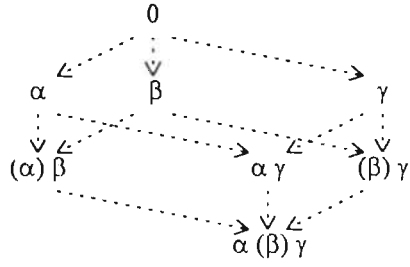

Figure 1: Reinstatement

The diagram shows that in only one of the lines of argumentation the argument α is reinstated, namely in the line of argumentation in which first α, second β, and third γ is taken into account.

In a line of argumentation, the status of an argument can change again and again. This leads to the notion of the status of an argument 'in the limit'. If in a line of argumentation from some stage onwards the status of an argument remains

constant, either undefeated or defeated, the argument is said to be undefeated or defeated in the limit, respectively.[30]

5.3 Definition of lines of argumentation and argumentation diagrams

Shortly we define lines of argumentation and argumentation diagrams. We need some auxiliary definitions.

In order to capture the six ways of argument construction that we discussed, we observe that they have a common characterizing property: the structure of the initial arguments is reflected in the structure of the newly constructed argument. The structural reflection of an argument in another is made precise in the following definition.

Definition 20.

The *maximal argument scheme* of an argument is defined recursively as follows:

1. The maximal argument scheme of an argument of the form *Sentence* is $*Sentence$.
2. The maximal argument scheme of the form $\{\{Argument_1, ..., Argument_n\} \rightarrow Conclusion$ is $\{\{MaxArgScheme_1, ..., MaxArgScheme_n\} \rightarrow Conclusion$, where $MaxArgScheme_i$ is the maximal argument scheme of $Argument_i$, for all i = 1, ..., n.
3. The maximal argument scheme of $\{Arguments_1, ..., Arguments_n\} \rightarrow Conclusion$ is $\{MaxArgSchemes_1, ..., MaxArgSchemes_n\} \rightarrow Conclusion$, where $\{MaxArgSchemes_i\} \rightarrow Conclusion$ is the maximal argument scheme of $\{Arguments_i\} \rightarrow Conclusion$, for all i = 1, ..., n.

An argument *Argument* is *structurally reflected* in an argument *Argument'* if there is an argument in the range of *Argument'* that is an instance of the maximal argument scheme of *Argument*.

The maximal argument scheme is just the argument itself, but with 'wildcarded premises'. The term 'maximal argument scheme' is used because the maximal argument scheme of an argument is the argument scheme that has a (the) maximal set of instances among the argument schemes that have the argument as an instance.

We can now define the successors of a stage, lines of argumentation and argumentation diagrams. The following definition implicitly refers to an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*).

---

[30] In Pollock's Theory of Defeasible Reasoning (Pollock, 1987-1995) and Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993) a similar notion is defined.

Definition 21.

A stage $Stage_1$ has a stage $Stage_2$ as its *successor* if all arguments in the range of $Stage_1$ are structurally reflected in a stage $Stage_2$. A *line of argumentation* is a sequence of argumentation stages $Stage_1$, $Stage_2$, ..., $Stage_n$, ... (not necessarily finite), such that for all natural numbers i $Stage_{i+1}$ is a successor of $Stage_i$. A *line of forward argumentation* is a line of argumentation that consists of stages with a constant set of premises. A *line of backward argumentation* is a line of argumentation that consists of stages with a constant set of conclusions. An *argumentation diagram* is a set of lines of argumentation.

In a line of argumentation, there is no constraint on the status of the arguments.[31] A stage can have zero, one or several successors. In fact, stages will often have many successors.

Our definitions of successors and lines of argumentation are related to Vreeswijk's definition of successors and argumentation sequences, respectively. However, they differ in three ways. First, the approaches to defeat in Vreeswijk's model and in CumulA are different (see chapter 6, section 4). Second, Vreeswijk's argumentation sequences represent how the set of arguments that are currently undefeated changes in argumentation, while CumulA's lines of argumentation represent how the set of arguments taken into account, whether undefeated or defeated, changes, and how the statuses of the arguments change. Third, CumulA's lines of argumentation are more general than Vreeswijk's argumentation sequences since the latter have fixed premises. Vreeswijk's argumentation sequences are therefore comparable to CumulA's forward lines of argumentation. The relation between successors in Vreeswijk's argumentation sequences is simpler than the relation between successor stages in CumulA's forward lines of argumentation. This is due to the fact that CumulA's stages are representations of all arguments currently taken into account, whether undefeated or defeated, while Vreeswijk's argument structures are only representations of the arguments currently undefeated. The advantages and disadvantages of the two approaches deserve further study.

We stress that the definition of stages above is not a constructive definition of the successors of a stage. It does provide a construction of the *arguments* in the ranges of the successor stages, but not of the *statuses* of these arguments. It is probably not easy to define the relation between the statuses of the arguments in the range of a stage and in the range of a successor, since a change of status of one argument can affect the status of a cascade of other arguments.

---

[31] Henry Prakken has pointed out to me that in cases of multiple stages with equal range, a constraint on the status of arguments seems appropriate. Since each of the multiple stages represents a choice of status, it seems natural that the choice should be kept constant in the successor stages. The problem is that the choice cannot always be kept constant. As a result, it should be made precise how the choice can be kept 'as constant as possible'. We leave this problem for future research.

Nevertheless, CumulA's lines of argumentation represent how argumentation with defeasible arguments proceeds. More precisely, they represent how argumentation *can* proceed, and not how such argumentation *should* proceed. We give an example of the distinction: both a line of argumentation in which counterarguments are systematically neglected, and one in which at each stage arguments are challenged by counterarguments that are newly taken into account, fit in the definition above. The second seems closer to how argumentation should proceed. However, a line of argumentation of both types can serve a purpose. A line of argumentation of the first type can help to find all arguments supporting a fixed point of view, while one of the second type can lead more efficiently to justified conclusions.

Which lines of argumentation are preferred with respect to specific purposes and standards, e.g., efficiency, can be regarded as constraints on lines of argumentation. Such constraints define argumentation protocols. Because of the generality of CumulA's lines of argumentation, it seems likely that very different protocols can be defined on them. Research on protocol in the context of argumentation with defeasible arguments has only recently started (see note 6), and is a promising direction of future research.

We finish this section with the definition of forward and backward extensions. Intuitively, a forward extension is the result of collecting as many arguments as possible from a given set of premises. A backward extension is the result of collecting as many arguments as possible, supporting a given set of conclusions.

Definition 22.

A *forward extension* of a set of sentences *Premises* is an argumentation stage *UndefeatedArguments* (*DefeatedArguments*) with premises in *Premises* that has no successor stage with premises in *Premises*. A forward extension *UndefeatedArguments* (*DefeatedArguments*) of a set of sentences *Premises* is *complete* if its range contains all arguments with premises in *Premises*. A *backward extension* of a set of sentences *Conclusions* is an argumentation stage *UndefeatedArguments* (*DefeatedArguments*) with conclusions in *Conclusions* that has no successor stage with conclusions in *Conclusions*. A backward extension *UndefeatedArguments* (*DefeatedArguments*) of a set of sentences *Conclusions* is *complete* if its range contains all arguments with conclusions in *Conclusions*.

A set of sentences can have zero, one, or several forward and backward extensions (possibly with empty range).

The definition of forward extensions has counterparts in many argumentation models, but the distinction between forward and backward extensions is to our knowledge new. Formally our definitions of extensions and complete extensions are close to Dung's (1993, 1995) preferred and stable extensions,

respectively.[32] Verheij (1996a) shows the formal relations between Dung's definitions and our definitions (for a version of CumulA, restricted to unstructured arguments and simple defeaters). It turns out that there are subtle distinctions and that the definition of extensions above corresponds somewhat better to the intuition that in an extension as many arguments are taken into account as possible.

6 Examples

In this section, we discuss a number of examples of argumentation theories in CumulA. The examples are meant as an illustration of the formal definitions of CumulA.

6.1 Sequential weakening and parallel strengthening

In the sections 3.3 and 3.4, we discussed examples of sequential weakening and parallel strengthening. Here we describe the argumentation theories corresponding to the examples.

First, we treat the sequential weakening example about heaps of sand, based on the sorites paradox. The following argumentation theory represents it, for a fixed natural number n:

Language = {Heap(i) | i = 0, 1, 2, ... }
Rules = {Heap(i) → Heap(i + 1) | i = 0, 1, 2, ... }
DefeaterSchemes = {[Heap(i)] | i > 0 }
∪ {[*Heap(i) → Heap(i+1) → ... → Heap(i+n)] | | i = 0, 1, 2, ... }[33]

Here Heap(0) abbreviates Body_of_sand_is_heap, Heap(1) abbreviates Body_of_sand_minus_1_grain_is_heap, and for each i = 2, 3, ..., Heap(i) abbreviates Body_of_sand_minus_i_grains_is_heap. The rules say that a body of sand that is one grain fewer than a heap is also a heap. The first set of defeater schemes represents that only the original body of sand is considered a heap without further argumentation. The second set of defeater schemes represents that any argument that contains a sequence of n steps of the rule is defeated.[34] The defeater

[32] Since Dung (1993, 1995) considers unstructured arguments, there is no distinction between forward and backward extensions.

[33] For convenience, we have left out the brackets { }.

[34] The choice of n determines the 'risk' we accept: for n not too large, say ten, in only a few cases a body of sand is wrongly judged a heap, but at the same time in a few cases reasoning can help us to determine that a body of sand is a heap. For n large, say a billion, we will more often wrongly judge a body of sand a heap, but also reasoning can help us more often. This trade-off between making mistakes and achieving the right results is paramount in reasoning with defeasible arguments.

exactly represents that such an argument becomes defeated because it contains too many steps.

The only statement that can be undefeated is

α_0: Heap(0)

Therefore the only arguments of this theory that might be undefeated are, for i = 0, 1, 2, ... :

α_i: Heap(0) → Heap(1) → ... → Heap(i)

If the arguments α_0, α_1, ... are consecutively taken into account, the resulting line of argumentation is the following sequence of stages:

α_0
$\alpha_0\ \alpha_1$
$\alpha_0\ \alpha_1\ \alpha_2$
...
$\alpha_0\ \alpha_1$ ... $\alpha_{n-1}\ (\alpha_n)$
$\alpha_0\ \alpha_1$ ... $\alpha_{n-1}\ (\alpha_n\ \alpha_{n+1})$
...

All arguments that contain a sequence of n steps are defeated. The first of these is the argument α_n. As a result, according to this theory, it is justified that the body of sand is a heap if at most n - 1 grains are taken away from the original heap.

If, for some natural number i_0, the conclusion Heap(i_0) could be justified by some other argument than α_{i_0}, the argument could be extended by n - 1 steps. It would be an undefeated argument different from the defeated α_{i_0+n-1}, and thereby justify that the original body of sand minus i_0 + n - 1 grains is a heap.

Second, we treat the parallel strengthening example about punishing John. The following argumentation theory represents it:

Language = {Robbed, Injured, Minor_first_offender, Punished, Not_punished}
Rules = {{Robbed} → Punished, {Injured} → Punished,
{Minor_first_offender} → Not_punished}
DefeaterSchemes =
{{{*Minor_first_offender}} → Not_punished [{{*Robbed}} → Punished],
{{*Minor_first_offender}} → Not_punished [{{*Injured}} → Punished],
{{*Robbed}, {*Injured}} → Punished
[{{*Minor_first_offender}} → Not_punished]}
∪ {[Punished], [Not_punished]}

The three rules say that John is punished if he has robbed, that John is punished if he has injured someone, and that John is not punished if he is a minor first

offender. The first two defeaters represent that any argument that ends in {{*Minor_first_offender}} → Not_punished rebuts any argument that ends in {{*Robbed}} → Punished or {{*Injured}} → Punished. The third defeater represents that any coordinated argument that ends with {{*Robbed}, {*Injured}} → Punished rebuts any argument that ends in {{*Minor_first_offender}} → Not_punished. The last two defeaters represent that the statements that John is punished and that John is not punished are defeated.

The following are the (non-statement) arguments of this theory:

α_1: {{Robbed}} → Punished
α_2: {{Injured}} → Punished
β: {{Minor_first_offender}} → Not_punished
α_{12}: {{Robbed}, {Injured}} → Punished

The arguments α_1 and α_2 are the narrowings of the argument α_{12}. In Figure 2, the main lines of argumentation with these arguments are shown.

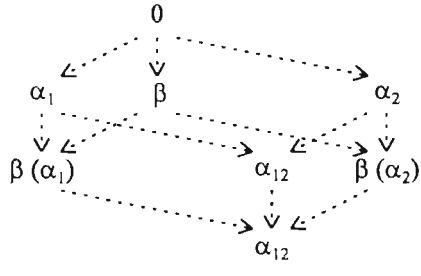

Figure 2: Parallel strengthening

The diagram shows that the arguments α_1 and α_2 only remain undefeated in a line of argumentation if they are both taken into account before β is.

6.2 Conflicting arguments: collective or multiple stages

It is often the case that arguments arise that have incompatible conclusions. Sometimes additional information can be used to resolve the conflict, for instance there can be information about the preference of the arguments.[35] However, it remains possible that there is not sufficient information to resolve the conflict. In that case, the conflict can be resolved by choosing one or more of the arguments involved in the conflict. Two general approaches to dealing with such situations

[35] In chapter 3, section 6, it is discussed how such conflict-resolving information can be represented in Reason-Based Logic. In chapter 4, section 5, other approaches of dealing with conflicts are treated.

have been proposed in the literature. The first is to discard all arguments in the conflict, as Pollock (1987) does, the second is to discard some of the arguments in such a way that the conflict is resolved, while as few arguments as possible are discarded, as for instance Vreeswijk (1991, 1993) does. Since in the latter case, there is normally no unique choice of arguments to discard, multiple solutions can arise.[36]

Both approaches have their merits, and seem reasonable in certain cases.[37] Therefore, in CumulA, both approaches can be dealt with, the first by collective defeat, and the second by multiple stages, i.e., different stages with equal range. As an example, we look at the following arguments:

John has stolen. So, he is punished.
John is a minor first offender. So, he is not punished.
It is nice to have a picnic in the woods. So, we go to the woods.
It is nice to have a picnic at the sea. So, we go to the sea.

The first two of these arguments have incompatible conclusions, the second two also. In the first conflict, it seems best to consider both arguments defeated without further information. In the second conflict, it can be argued that one of the two arguments should be defeated, each choice being equally good. Both are modeled in the following argumentation theory (*Language*, *Rules*, *DefeaterSchemes*):

Language = {Has_stolen, Is_punished, Minor_first_offender, Is_not_punished, Nice_in_the_woods, Go_to_the_woods, Nice_at_sea, Go_to_the_sea}
Rules = {Has_stolen → Is_punished, Minor_first_offender → Is_not_punished, Nice_in_the_woods → Go_to_the_woods, Nice_at_sea → Go_to_the_sea}
DefeaterSchemes = {[*Is_punished, *Is_not_punished], *Go_to_the_woods [*Go_to_the_sea], *Go_to_the_sea [*Go_to_the_woods]}

The main arguments of this theory are:

---

[36] These solutions correspond to what are often called extensions. In the literature, three perspectives on multiple extensions have been proposed, as Makinson (1994, p. 38) notes: the skeptical perspective, the liberal (or credulous) perspective, and the choice perspective. The skeptical perspective focuses on the intersection of the extensions, the liberal perspective on their union, and the choice perspective on a selected extension. In CumulA, we prefer the latter perspective since the skeptical perspective is closely related to collective defeat, as Pollock (1992, p.7) remarks, which can be dealt with using a compound defeater (cf. section 3.5), while the liberal perspective does not help to resolve conflicts: the union of the multiple extensions that arise to resolve some conflict, again contains the conflict.

[37] For instance, Pollock (1994; 1995, pp. 62-64) argues that while in epistemic reasoning unjustified choices are unreasonable, in practical reasoning it is sometimes better to make some choice than none. Since he focuses on epistemic reasoning, he prefers the collective defeat approach.

α: Has_stolen → Is_punished
β: Minor_first_offender → Is_not_punished
γ: Nice_in_the_woods → Go_to_the_woods
δ: Nice_at_sea → Go_to_the_sea

The two conflicts are handled in different ways: the conflict of the arguments α and β is dealt with by the compound defeater [*Is_punished, *Is_not_punished], while the conflict of the arguments γ and δ is dealt with by two simple defeaters, *Go_to_the_woods [*Go_to_the_sea] and *Go_to_the_sea [*Go_to_the_woods].

Figure 3 shows two argumentation diagrams of this theory. On the left, the arguments α and β are taken into account, and are collectively defeated. On the right, γ and δ are taken into account, resulting in two stages with the same range. (They are separated by a comma.) There are two stages with all four arguments as range, namely (α β) γ (δ) and (α β γ) δ.

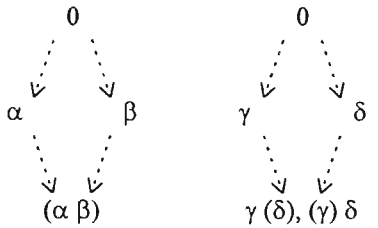

Figure 3: Collective defeat and multiple stages

Although the example argumentation theory is tailor-made for the four mentioned arguments, it shows how general argumentation theories can be defined, in which there is one class of arguments that are collectively defeated in cases of conflict, and another class of arguments that lead to multiple stages in cases of conflict.

To finish the example of collective defeat and multiple stages, we show what happens if there are additional arguments that challenge one of the arguments in the conflict. For instance, there might be two additional arguments

ε: Severe_crime
ζ: Stormy_weather

and two additional defeaters

*Severe_crime [*Minor_first_offender → Is_not_punished]
*Stormy_weather [*Go_to_the_sea].

In the case of collective defeat and in the case of multiple stages, one of the arguments involved in the conflict is reinstated. Taking into account the argument ε that John's crime was severe, has the effect that α, challenged by ε, is defeated,

and that as a result α and β are not collectively defeated. Taking into account the argument ζ that the weather is stormy, has the effect that γ, challenged by ζ, is defeated, and that as a result γ and δ do not give rise to multiple stages. Figure 4 shows the corresponding argumentation diagrams.

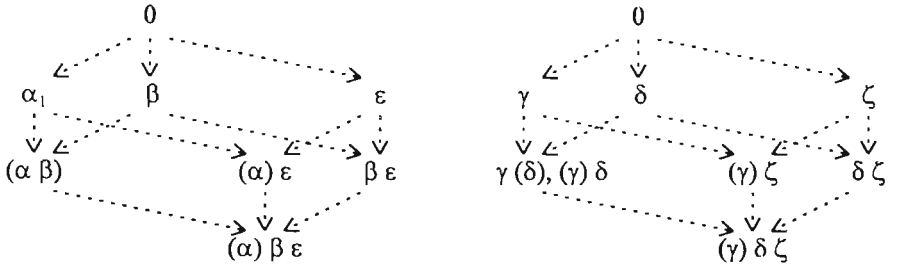

Figure 4: Reinstatement of conflicting arguments

The diagrams shows that collective defeat and multiple stages still occur if ε and ζ are not taken into account.

6.3 Stable marriages

Dung (1995) discusses the so-called *stable marriage problem* in terms of argumentation. In this problem, there is a number of people, some of which love someone else, and some of which are married or, more generally, have a love affair. However love is not always answered, and people do not always have a love affair with the one they love. As a result, love affairs are not necessarily stable. For instance, if John loves Mary, and Mary has a love affair with Alex, the affair of Mary and Alex is in danger, since John will strive for an affair with Mary. However, this threat to Mary and Alex's love affair is overcome if Mary loves Alex: in that case, she will not answer John's attempts. The problem is now to determine which collections of love affairs are stable.[38]

We examine the case that there is a 'love circle': there are n persons $person_1$, ..., $person_n$ (with n larger than 2), and for i = 1, ..., n, $person_i$ loves $person_{i+1}$, and $person_n$ loves $person_1$. In this situation, the fact that $person_i$ loves $person_{i+1}$ is a threat to the affair which $person_{i+1}$ has with $person_{i+2}$. This case can be translated to an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*), as follows.

[38] Dung (1995) discusses the slightly more general problem, in which each person has linearly ordered the other persons according to his or her 'love preference'.

Language = {Loves($person_i$, $person_{i+1}$), Affair($person_i$, $person_{i+1}$) | i is an integer modulo n}[39]
Rules = {Loves($person_i$, $person_{i+1}$) → Affair($person_i$, $person_{i+1}$) | i is an integer modulo n}
DefeaterSchemes = {Loves($person_i$, $person_{i+1}$) [*Affair($person_{i+1}$, $person_{i+2}$)] | i is an integer modulo n}

We consider the following arguments, for i an integer modulo n:

α_i: Loves($person_i$, $person_{i+1}$) → Affair($person_i$, $person_{i+1}$)

These arguments represent that if $person_i$ loves $person_{i+1}$, $person_i$ strives for an affair with $person_{i+1}$.

In the case there are four persons (i.e., n = 4), there are two stable situations, in which all four persons have an affair: either $person_1$ and $person_2$ have an affair, and $person_3$ and $person_4$ have an affair, or $person_2$ and $person_3$ have an affair, and $person_1$ and $person_4$ have an affair. Figure 5 shows the resulting argumentation diagram, for n = 4, that ends in two stages with equal range, that correspond to the two intuitive solutions.

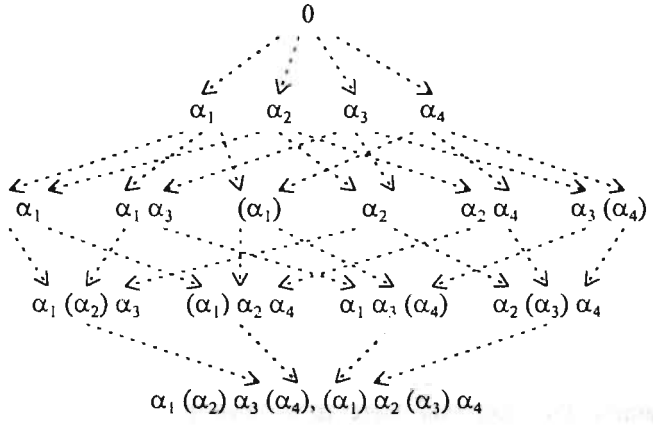

Figure 5: The four-persons case

In the three-persons case (i.e., n = 3), there is no stable solution: any love affair will be threatened.[40] This instability is reflected in the corresponding argumentation diagram (Figure 6). It turns out that there is no stage in which all three arguments are taken into account. Any pair of arguments can be taken into

[39] Here 'i modulo n' means 'the remainder of the integer division i/n'.
[40] Note that for n odd at least one of the love affairs involves two persons of the same sex.

account, but the third argument cannot be. In the figure this is indicated by three question marks ???.[41]

The stages in Figure 6 have a meaning in terms of argumentation. For instance, in the stage α_1 (α_2) the argument α_1 is not challenged, since the argument α_3 is not yet taken into account. The argument α_2 is challenged by Loves(person$_1$, person$_2$). As a result, α_1 justifies Affair(person$_1$, person$_2$), while α_2 cannot justify Affair(person$_2$, person$_3$).

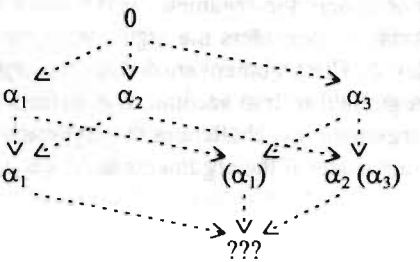

Figure 6: The three-persons case

The three-persons and four-persons cases directly generalize to the cases of any odd and even number of persons, respectively. In the odd case, there is no overall stable solution, in the even case there are two.

6.4 The neurotic fatalist

In the three-persons case above, we saw that not all ranges of argumentation theories correspond to a stage. However, in that example there were maximal subranges that did correspond to a stage, viz. the two-argument subranges. We now show an argumentation theory that has a range, such that there is no maximal subrange that corresponds to a stage.

As an example, we consider the story of the neurotic fatalist. There is one thing our fatalist has been certain of for months: if the world does not end today, it will end tomorrow. Each morning after sunrise he admits that he was wrong the day before, and that the world does not yet end today, but that he nevertheless believes that the world will end the next day.

The arguments of the neurotic fatalist can be formalized in the following argumentation theory:

---

[41] The fact that there is no stage with all three arguments corresponds in Dung's (1995) approach to the fact that there is no stable extension. The stage approach gives more information about the argumentation theory than Dung's approach since there are stages with less than three arguments. See Verheij's (1996a) comparison of the two approaches for details.

Language = {World_ends(day$_i$), ¬World_ends(day$_i$) | i = 0, 1, 2, ... }
Rules = {¬World_ends(day$_i$) → World_ends(day$_{i+1}$) | i = 0, 1, 2, ... }
DefeaterSchemes = {*¬World_ends(day$_i$) [*World_ends(day$_j$)] | i > j }

We consider the following arguments, for i any natural number:

α_i: ¬World_ends(day$_i$) → World_ends(day$_{i+1}$)

At day 0, our fatalist considers the argument α_0 that the world ends at day 1. It is undefeated. The next day he considers the argument α_1: the world did not end at day 1, so it ends at day 2. The argument α_1 defeats the argument α_0. At each new day, he takes a new argument α_i into account, that defeats all previous arguments, since, for each i, the argument α_{i+1} challenges the argument α_i.

We get the following stages if the arguments α_0, α_1, α_2, α_3, ... are consecutively taken into account:

α_0
$(\alpha_0)\ \alpha_1$
$(\alpha_0\ \alpha_1)\ \alpha_2$
$(\alpha_0\ \alpha_1\ \alpha_2)\ \alpha_3$
. .

In Figure 7, an overview of these stages is given in an argumentation diagram of the theory.

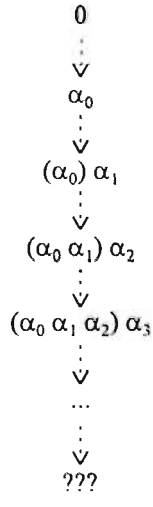

Figure 7: The case of the neurotic fatalist

Although the argumentation theory itself may not be considered sensible, the theory is technically interesting since there is no stage with all arguments α_i in its range, nor a maximal subrange that corresponds to a stage. Nevertheless there are several sensible stages. This can be seen as follows.

Assume first that there is a maximal subrange *Subrange*. If *Subrange* is finite, there is a natural number i_0 that is the maximum of the indices i of the arguments α_i in *Subrange*. But then the stage $(\alpha_1\ \alpha_2\ \ldots\ \alpha_{i_0})\ \alpha_{i_0+1}$ has larger range, which contradicts the assumption. Therefore we can assume that *Subrange* is infinite. It is impossible that all arguments α_i in *Subrange* are defeated, since in this argumentation theory an argument can only be defeated by an undefeated argument. Therefore, let i_0 be the smallest natural number i, such that α_i is not defeated. However, if α_{i_0} is not defeated, all arguments that challenge it, i.e., all α_i with $i > i_0$, must be defeated. But that is impossible, since then for each argument α_i there must be an undefeated argument that challenges α_i, and such an argument must have an index larger than i. This contradicts the choice of i_0.

Chapter 6

Analyzing argumentation models using CumulA

After the description of the argumentation model CumulA in chapter 5, we show how CumulA can be used to analyze existing argumentation models. We start with a discussion of distinctions that can be made between argumentation models. We make these distinctions precise by showing their formal counterparts for CumulA's argumentation theories. After capturing elements of a number of existing argumentation models in CumulA's argumentation theories, we apply the distinctions to these argumentation theories.

In section 1, we discuss types of arguments. In section 2, we treat argument structure and defeat. We distinguish sentence-type, step-type and composite-type defeat. In section 3, we consider individual and groupwise defeat. In section 4, we characterize triggers of defeat. We distinguish inconsistency-triggered and counterargument-triggered defeat. In section 5, we deal with directions of argumentation. We distinguish forward, backward and bidirectional argumentation. In section 6, we capture elements of several major argumentation models in CumulA's argumentation theories.[1] In section 7, the distinctions made are applied to these argumentation theories. In this way, the argumentation theories capturing elements of existing argumentation models can be compared on formal grounds.

1 Types of arguments

Several types of arguments, that have been proposed in argumentation models, can in CumulA (chapter 5) be distinguished by their structure.

The first type of arguments are the *statements*, that have trivial structure:

Statement.

[1] We stress that we give no formal relations between the argumentation models and CumulA's argumentation theories.

Many argumentation models do not deal with structured arguments. For instance, Poole's Logical Framework for Default Reasoning (Poole, 1988)[2] uses special sets of sentences without structure. In Dung's Argumentation Frameworks (Dung, 1993, 1995), arguments are structureless objects, that can attack each other.

The second type of arguments are the *single-step arguments*, which have the simplest non-trivial structure:

Reason.
So, *Conclusion*.

For instance, in Propositional and First-Order Predicate Logic,[3] the semantical and proof-theoretical consequence relations, denoted as $\models$ and $\vdash$, respectively, which are often interpreted as arguments (e.g., Purtill, 1979; Copi, 1982), have this structure.

The third type of arguments are the arguments that are constructed by *subordination*, such as the argument:

Reason$_1$.
So, *Reason*$_2$.
So, *Conclusion*.

This argument structure is most common. For instance, in Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993),[4] arguments are explicitly constructed by subordination. Also the proofs of several proof theories for Propositional or First-Order Predicate Logic have this structure. Less obviously, this structure is also at the heart of Reiter's Default Logic (Reiter, 1980, 1987),[5] Bondarenko *et al.*'s Assumption-Based Framework for Non-Monotonic Reasoning (Bondarenko *et al.*, 1993), and Loui and Chen's Argument Game (Loui and Chen, 1992). Pollock's linear arguments in his Theory of Defeasible Reasoning (1995, p. 39)[6] can be regarded as having this structure.[7]

---

2 See also chapter 4, section 4.2.

3 See, e.g., Van Dalen (1983) or Davis (1993).

4 See also chapter 4, section 5.2.

5 See also chapter 4, sections 3.1, 4.2 and 5.2.

6 See also chapter 4, section 4.2.

7 Pollock (1995, p. 39) defines linear arguments as finite sequences of sentences, each of which is either a premise or supported by a previous member of the sequence. The structure of linear arguments is not only ambiguous, as Pollock (1995, p. 87) notes, but is somewhat less expressive than that of subordinated arguments, because it cannot distinguish different occurrences of the same sentence in an argument. For instance, the arguments $\{\{\{\{A\}\} \rightarrow B\}\} \rightarrow C$ and $\{\{\{\{A\}\} \rightarrow B,\ A\}\} \rightarrow C$ in CumulA both correspond to the linear argument A, B, C.

The fourth type of arguments are the arguments that are constructed by both *subordination and coordination* of arguments, for instance:

> $Subreason_{11}$, $Subreason_{12}$; $Subreason_{21}$, $Subreason_{22}$.
> So, *Conclusion*.

This is the argument structure that is used in CumulA. In the argumentation theory of Van Eemeren and Grootendorst (Van Eemeren *et al.*, 1981, 1987), real-life arguments are reconstructed and evaluated using the mentioned argument structure.[8] Van Eemeren and Grootendorst have included both subordination and coordination in their model since both can be found in argumentative texts. In the next section, we argue for the need of coordination, especially for defeasible arguments because of defeat by pararallel strengthening and the accrual of reasons.

We mention a fifth type of argument structure that occurs, for instance, in natural deduction proofs of Propositional and First-Order Predicate Logic, and in Pollock's Theory of Defeasible Reasoning (Pollock, 1987-1995): *arguments with suppositions*. For instance, such arguments occur if the natural deduction rule of inference →-Introduction is used in a proof or argument:

> A proof of Q with suppositions in a set $S \cup \{Q\}$ can be extended to a proof of $P \rightarrow Q$ with suppositions in the set S.

Here, a proof is considered relative to a set, the suppositions of the proof. The rule of inference→-Introduction above shows that the set of suppositions can change. After the introduction of $P \rightarrow Q$, the supposition Q can be withdrawn.

If one reads 'argument' instead of 'proof', this rule of inference becomes a type of argument construction, as Pollock does. To include this type of argument construction in his argumentation model, Pollock (1995, p. 86ff.) constructs arguments not from sentences (as in CumulA), but from sentences relative to a set of suppositions, formally an ordered pair of a sentence and a set of sentences (P, S). For instance, the rule of inference →-Introduction becomes: [9]

> An argument supporting (Q, $S \cup \{Q\}$) can be extended to an argument supporting ($P \rightarrow Q$, S).

We have not included this type of argument in CumulA for two reasons. First, we think that the intuition of an argument without suppositions is easier to grasp than the intuition of an argument with suppositions. Whereas arguments without suppositions can be thought of as consisting of steps that represent the support of a

---

[8] The terminology of Van Eemeren and Grootendorst differs from ours. Their multiple arguments correspond to CumulA's coordinated arguments (cf. chapter 5, note 7).

[9] We paraphrase Pollock's 'rule of inference graph formation' called conditionalization (Pollock, 1995, p. 90).

state of affairs (expressed by a sentence) by another state of affairs (expressed by another sentence), arguments with suppositions cannot be thought of that way. This is due to the fact that some

> '... natural deduction rules have an indirect, even quasi-metalogical character' (Haack, 1978, p. 19).

This does of course not diminish the importance of the arguments with suppositions based on natural deduction rules, and their role in argumentation certainly deserves further study.

Second, arguments with suppositions behave unexpectedly if they are defeasible, as Vreeswijk (1993, p. 185ff.) shows. He gives a technical example in which arguments that should be undefeated nevertheless become defeated if the rule of $\rightarrow$-Introduction is adopted. Vreeswijk's conclusion is that it is best to leave arguments with suppositions out of theories of argumentation with defeasible arguments for now until we have a better understanding of the behavior of more simply structured defeasible arguments. Since, to the best of our knowledge, the problems pointed out by Vreeswijk have not been solved, we have adopted the same conclusion.

2 Argument structure and defeat

The structure of an argument can determine whether an argument is defeated. In this section, we treat different types of structure-based defeat, as they are found in existing argumentation models. We show how the types of defeat can be distinguished in CumulA.

The first and simplest type of structure-based defeat is the trivial type of *no defeat* at all. The prototypical examples of argumentation models that have no defeat are the classical deductive logics, such as Propositional and First-Order Predicate Logic. In CumulA, an argumentation theory has no defeat if it has no defeater schemes.

The second type of structure-based defeat is *sentence-type defeat*. The defeat of an argument is of sentence-type if the defeat depends on sentences occurring in the argument. For instance, an argument

Reason.
So, *Conclusion.*

might be defeated because of an (undefeated) statement that denies the conclusion, such as:

Not_conclusion.

This is a case of sentence-type defeat: any argument containing the sentence *Conclusion* is defeated if the statement *Not_conclusion* is undefeated. A defeater scheme representing this in CumulA has the form

Not_conclusion [**Conclusion*].

The challenged argument scheme **Conclusion* has any argument with conclusion *Conclusion* as an instance. If any argument with conclusion *Not_conclusion* challenges any argument with conclusion *Conclusion*, this would be represented by the defeater scheme

Not_conclusion* [Conclusion*].

We say that the two mentioned defeater schemes are of sentence-type, which means that all their argument schemes have a statement as an instance. An argumentation theory has sentence-type defeat if it has sentence-type defeater schemes.

Argumentation models with sentence-type defeat are Poole's Logical Framework for Default Reasoning (Poole, 1988), and Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993). Also Dung's Argumentation Frameworks (Dung, 1993, 1995) can be regarded as having sentence-type defeat since all arguments are structureless.

Bondarenko *et al.*'s Assumption-Based Framework for Non-Monotonic Reasoning (Bondarenko *et al.*, 1993) describe a special kind of sentence-type defeat, that we call *assumption-type defeat*. There is a special set of assumptions, that can be used as premises of arguments. If there is an undefeated argument that has the denial of an assumption as its conclusion, all arguments with that assumption as a premise are defeated. A defeater scheme representing this in CumulA has the form

**Not_assumption* [*Assumption*].

This defeater scheme has no consequences for arguments that do not have *Assumption* as a premise, even if *Assumption* occurs in the argument elsewhere. A sentence-type defeater scheme, as the one above, that has only statements as challenged arguments, is of assumption-type. An argumentation theory has assumption-type defeat if it has defeater schemes of assumption-type.

The third type of structure-based defeat is *step-type defeat*. The defeat of an argument is of step-type if the defeat depends on a step occurring in the argument. For instance, an argument

Reason.
So, *Conclusion*.

might be defeated because there is an (undefeated) statement that does not deny the conclusion, but undercuts the argument step (cf. chapter 5, section 3.1):

Undercutter

This is a case of step-type defeat: any argument containing the argument step '*Reason*. So, *Conclusion*' is defeated if the conclusion *Undercutter* is justified. A defeater scheme representing this in CumulA has the following form:

$*Undercutter\ [\{\{*Reason\}\} \rightarrow Conclusion]$.

Another example of step-type defeat is rebuttal (cf. chapter 5, section 3.2): an argument

Reason$_1$.
So, *Conclusion*.

is defeated because there is an (undefeated) argument that supports the denial of its conclusion:

Reason$_2$.
So, *Not_conclusion*.

Any argument containing the step '*Reason*$_1$. So, *Conclusion*' is defeated if an argument containing the step '*Reason*$_1$. So, *Conclusion*' is undefeated. A defeater scheme representing this in CumulA has the following form:

$\{\{*Reason_2\}\} \rightarrow Not\_conclusion\ [\{\{*Reason_1\}\} \rightarrow Conclusion]$

The latter two defeater schemes are of step-type: all their argument schemes have a single-step argument as an instance that is not of sentence-type. An argumentation theory has step-type defeat if it has step-type defeater schemes.

The fourth type of structure-based defeat is *composite-type defeat*. We speak of composite-type defeat if the defeat of an argument depends on a composite structure occurring in the argument. In chapter 5, sections 3.3 and 3.4, we discussed two kinds of composite-type defeat: defeat by sequential weakening and defeat by parallel strengthening. We recall that in defeat by sequential weakening an argument is defeated because it ends in some sequence of steps. A defeater scheme representing that any argument ending with the two-step sequence '*Reason*. So, *Conclusion*$_1$. So, *Conclusion*$_2$' is always defeated has the following form:

$[\{\{\{\{*Reason\}\} \rightarrow Conclusion_1\}\} \rightarrow Conclusion_2]$

In defeat by parallel strengthening an argument is defeated because some argument that has narrowings (chapter 5, section 2.4) is undefeated. A defeater representing that any argument in which two reasons $Reason_1$ and $Reason_2$ support the conclusion *Conclusion* defeats any argument in which the reason $Reason_3$ supports *Not_conclusion* has the following form:

$$\{\{*Reason_1\}, \{*Reason_2\}\} \rightarrow Conclusion\ [\{\{*Reason_3\}\} \rightarrow Not\_conclusion]$$

The latter two defeaters are of composite-type, meaning that they are neither of sentence-type nor of step-type.[10] An argumentation theory has composite-type defeat if it has composite-type defeater schemes.

Most existing argumentation models do not have composite-type defeat. An exception is Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993). In Vreeswijk's formalism defeat depends on a conclusive force relation on full arguments. However, since Vreeswijk only uses subordination to construct composite arguments and no coordination, his formalism only can model defeat by sequential weakening and not defeat by parallel strengthening.

Defeat by parallel strengthening requires the coordination of arguments. It is based on the natural idea of accrual of reasons:[11] A conclusion can be better supported if there are more independent reasons for it. Although several people have made the point that reasons can accrue,[12] it remains controversial.

For instance, Pollock (1991a, 1995, pp. 101-102) explicitly argues against accrual. He thinks accrual is a natural idea, but then gives an example that makes him doubt that reasons accrue. The example goes as follows. If someone testifies that the president of Slobovia has been assassinated, that is a reason that the president is assassinated. Accrual would imply that testimonies of different people make the fact that the president is assassinated more credible. Pollock points out that this does not generally hold and depends on contingent facts. For instance, if testimonies are indeed independent, they make the president's assassination more credible. However, the testimonies are not necessarily independent: we can imagine a community in which people only confirm each other's lies. In that case, more reasons based on testimonies do not give increasing support to the president's assassination: more than one testimony would even make the assassination unjustified.[13]

---

[10] Defeater schemes of composite-type should not be confused with compound defeater schemes. Compound defeater schemes are defeater schemes that contain more than one challenging or more than one challenged argument scheme (chapter 5, sections 3.5 and 3.7). See also the next section on individual and groupwise defeat.

[11] Pollock (1991, p. 51) uses this terminology.

[12] Chronologically: Naess (1978) in argumentation theory, Hage (1991) in legal reasoning, Pinkas (1991) in neural computing, Brewka and Gordon (1994) and Gabbay (1994, pp. 196-198) in formal logic, Visser (1995, p. 177) in AI and law.

[13] A similar, more realistic, example is the following, by Henry Prakken. John likes to walk if it is Sunday. John does not like to walk if it is either hot or raining. If it is either hot

As a solution, Pollock proposes that different independent reasons for a conclusion are subsumed in a new composite reason. In our opinion, this approach probably can be made to work - Pollock does not give details. However, the example does not *necessitate* Pollock's approach, while the approach does throw away the intuitively attractive idea of accrual of reasons. Both in chapter 2 on Reason-Based Logic and in chapter 5 on CumulA, we have presented formalisms that capture accrual and still can deal with examples such as Pollock's. For instance, Pollock's example is captured in CumulA by the following compound defeater scheme:

$[\{\{*\text{Testimony}_1\}, \{*\text{Testimony}_2\}\} \rightarrow \text{Assassination}]$

Moreover, properties characteristic for accrual, such as the property that if a narrowing of an argument is undefeated, the argument itself is undefeated (chapter 5, section 4.1), and the property that, if the pros outweigh the cons, additional pros do not change the balance (chapter 2, section 5), can easily be overlooked.

3 Individual and groupwise defeat

The defeat of an argument often depends on other arguments. Mostly the defeat of an argument depends on one other argument, but not always. In this section, we distinguish argumentation models by the number of arguments that determine defeat.

First, the defeat of an argument can depend only on itself, and not on any other argument. We call this *self-defeat*. For instance, an argument that has a contradiction as its conclusion often is considered defeated, for instance in Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993). In CumulA, this could be represented by a defeater scheme of the following form:

$[*\textit{Contradiction}]$

Another example is an argument that is defeated because it contains some sequence of steps, as in defeat by sequential weakening (chapter 5, section 3.3). If an argumentation theory has defeater schemes, the instances of which have no challenging and one challenged argument, we say the argumentation theory has self-defeat.

---

or raining on Sunday, he does not like to walk. If it is hot and raining on Sunday, he likes to walk. The difficulty is here that the reasons 'It is hot' and 'It is raining' together are apparantly weaker, in contrast with the principle of accrual. Since we choose to keep the intuitively attractive principle of accrual, we propose to deal with this example by considering 'It is hot and raining' as a *new* reason, and not only as the coordination of two reasons.

Second, the defeat of an argument can depend on one other undefeated argument. We call this *simple defeat*. Examples are arguments that are defeated by an undercutter or by a rebutter, as distinguished in Pollock's Theory of Defeasible Reasoning (Pollock, 1987-1995). In CumulA, defeat by an undercutter or rebutter is represented by defeater schemes, such as the following two:

$*Undercutter\ [\{\{*Reason\}\} \rightarrow Conclusion]$
$\{\{*Reason_2\}\} \rightarrow Not\_conclusion\ [\{\{*Reason_1\}\} \rightarrow Conclusion]$

Both defeater schemes are simple since their instances have at most one challenging and at most one challenged argument (chapter 5, section 3.7). If an argumentation theory has simple defeater schemes, we say it has simple defeat.

Third, the defeat of an argument can depend on more than one undefeated argument. We call this *left-compound defeat* (because of the form of the corresponding defeater schemes). An example is an argument that is defeated because its conclusion conflicts with the conclusion of other arguments, as for instance in Poole's Logical Framework for Default Reasoning (Poole, 1988) and Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993). If $Conclusion_1$, ... $Conclusion_{n-1}$ and $Conclusion_n$ are conflicting, this can in CumulA be represented by a defeater scheme of the following form:

$*Conclusion_1, \ldots, *Conclusion_{n-1}\ [*Conclusion_n]$

This defeater scheme is left-compound since its instances have more than one challenging argument (chapter 5, section 3.7). If an argumentation theory has left-compound defeater schemes, we say it has left-compound defeat.

Fourth, the defeat of an argument can depend on other defeated arguments. We call this *right-compound defeat*. An example is an argument that is defeated together with other arguments because their conclusions are conflicting, as the collective defeat of arguments in Pollock's Theory of Defeasible Reasoning (Pollock, 1987-1995). If $Conclusion_1$, ... $Conclusion_{n-1}$ and $Conclusion_n$ are conflicting, this can in CumulA be represented by a defeater scheme of the following form:

$[*Conclusion_1, \ldots, *Conclusion_{n-1}, *Conclusion_n]$

This defeater is right-compound since its instances have more than one challenged argument (chapter 5, section 3.7). If an argumentation theory has right-compound defeater schemes, we say it has right-compound defeat.

4 Triggers of defeat

Argumentation models can differ in the way the defeat of arguments is triggered. Two triggers of defeat can be distinguished: inconsistency and counterarguments. We call the resulting types of defeat inconsistency-triggered and counterargument-triggered defeat, respectively.[14]

Inconsistency-triggered defeat has the longest tradition and is related to the early work on nonmonotonic reasoning. Its basic intuition is that the defeat of arguments is at heart the maintenance of the consistency of argument conclusions. Many variants have been proposed. For instance, one of a (minimal) set of arguments with conflicting arguments can be considered defeated, as in Poole's Logical Framework for Default Reasoning (Poole, 1988) and Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993). If $Conclusion_1$, ... $Conclusion_{n-1}$ and $Conclusion_n$ are conflicting, this can in CumulA be represented by n (left-compound) defeater schemes of the following form:

$*Conclusion_1, \ldots, *Conclusion_{i-1}, *Conclusion_{i+1}, \ldots, *Conclusion_n$
$[*Conclusion_i]$

This leads to indeterministic defeat since each of these defeaters represents an arbitrary choice of a defeated argument (chapter 5, section 3.5).[15] In Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993), the choice of a defeated argument is restricted by a conclusive force relation: an argument in a minimal set of arguments with conflicting conclusions cannot be considered defeated if it has stronger conclusive force than one of the other arguments in the set.

If an argumentation theory has defeater schemes the instances of which consist of arguments with conflicting conclusions (with respect to some appropriate sense of inconsistency), we say the argumentation theory has inconsistency-triggered defeat.

Counterargument-triggered defeat is based on another intuition: defeat is the result of arguments challenging other arguments. The purest version of counterargument-triggered defeat is Dung's formalism of Argumentation Frameworks (Dung, 1993, 1995). Dung studies a binary attack relation between arguments. In CumulA, his attacks can be represented as defeaters of the following form:

$Argument_1$ [$Argument_2$]

---

[14] The distinction between inconsistency-triggered and counterargument-triggered defeat corresponds to Verheij's (1995a, b) distinction between indirect and direct defeat.

[15] As a result, indeterministic defeat leads to multiple extensions, as in many models of nonmonotonic reasoning. Cf. the overviews by Ginsberg (1987), Lukaszewicz (1990) and Gabbay *et al.* (1994b). See also chapter 5, section 6.2.

Attacks are represented as defeaters and not as defeater schemes since Dung treats arguments as structureless objects. As a result his arguments correspond to statements in CumulA. If an argumentation theory has defeater schemes the instances of which do not consist of arguments with conflicting conclusions (with respect to some appropriate sense of inconsistency), we say the argumentation theory has counterargument-triggered defeat. Clearly, general argumentation theories in CumulA have counterargument-based defeat.

In a way, counterargument-triggered defeat is more general than inconsistency-triggered defeat. Whereas inconsistency-triggered defeat can naturally be captured as a special case of counterargument-based defeat (as in the examples above), not all counterargument-triggered defeat can as naturally be captured as a special case of inconsistency-triggered defeat.

The distinction between inconsistency-triggered and counterargument-based defeat can be recognized if one considers rebutting and undercutting defeat. Rebutting defeat is by its nature an example of inconsistency-triggered defeat, but can as we have seen naturally be captured in the defeater schemes CumulA, which has counterargument-triggered defeat. Undercutting defeat is by its nature an example of counterargument-triggered defeat, and can naturally be captured in CumulA's defeater schemes, but not as naturally in inconsistency-triggered defeat.

For instance, Vreeswijk (1993, pp. 51-53) claims that it is possible to incorporate undercutting defeat in his Abstract Argumentation Systems, which have inconsistency-triggered defeat. However, in order to incorporate undercutting defeat, Vreeswijk has to adapt his argumentation model, as follows. He introduces a defeasible conditional $>$ in his language. In a case of undercutting defeat, Vreeswijk forces an inconsistency between the conditional and its negation. The use of defeasible conditionals is a fine approach to undercutting defeat, and is very similar to the approach of Reason-Based Logic (chapter 2), but requires an adaptation of the formalism. Moreover, Vreeswijk hinges on two thoughts: he incorporates undercutting defeat using defeasible conditionals and rebutting defeat using argument defeat. However, we have seen that it is possible to capture both undercutting and rebutting defeat using defeasible conditionals (as for instance in Reason-Based Logic), and using argument defeat (as for instance in CumulA).

5 Directions of argumentation

Argumentation models can differ in the direction of argumentation they describe. We distinguish static, forward, backward and bidirectional argumentation.

Static argumentation occurs in argumentation models that do not treat argumentation as a process. No sequences of stages are considered, but only stages that are in some sense maximal. The extensions of Reiter's Default Logic (Reiter, 1980, 1987) and Poole's Logical Framework for Default Reasoning (Poole, 1988) can be regarded as such special stages.

Forward argumentation is the most common among existing argumentation models. Argumentation starts from a fixed set of premises. Arguments are constructed by adding forward steps. In forward argumentation, the goal is to find conclusions supported by arguments with given premises. For instance, Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993), Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993) and Bondarenko *et al.*'s Assumption-Based Framework for Non-Monotonic Reasoning (Bondarenko *et al.*, 1993) are models of forward argumentation. In CumulA, forward argumentation means that a line of argumentation only contains stages with premises in a fixed set.

Backward argumentation is less common. Argumentation starts from a set of conclusions. Arguments are constructed by adding backward steps. In backward argumentation, the goal is to find premises for arguments supporting given conclusions. For instance, Loui and Chen's Argument Game (Loui and Chen, 1992)[16] is a model with backward argumentation. In CumulA, backward argumentation means that a line of argumentation only contains stages with conclusions in a fixed set.

Bidirectional argumentation is the natural generalization of forward and backward argumentation. Argumentation does not start form a fixed set of premises or conclusions. Arguments are both forwardly and backwardly constructed. In bidirectional argumentation, the goal is neither only to find conclusions nor only to find premises, but a mixture of both. Except for CumulA, we know of no argumentation model of bidirectional argumentation.[17]

6 Capturing elements of argumentation models in CumulA

In the previous sections, we have discussed several ways to distinguish argumentation models. We explained how these distinctions can be made for CumulA argumentation theories. To be able to use the distinctions to compare existing argumentation models, we show how elements of a number of major argumentation models can be captured in argumentation theories of CumulA. We stress that we do not give formal relations between argumentation models and CumulA's argumentation theories. The presented argumentation theories capturing elements of existing argumentation models are meant to illustrate CumulA and our views on other argumentation models, and not to show strict formal relations.

Our selection of argumentation models is influenced by our focus, as made explicit by the CumulA model. Each selected argumentation model has been influential, or shows a specific characteristic of argumentation that falls within our

[16] Recently, a variant of Loui and Chen's Argument Game has been implemented by Kang.

[17] Pollock (1995, p. 153) describes forward and backward argumentation in another sense: he keeps both allowed premises and desired conclusions fixed. In bidirectional argumentation in our sense, neither premises nor conclusions are fixed.

focus. We have selected Propositional Logic, Poole's Logical Framework for Default Reasoning (Poole, 1988), Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993), Reiter's Default Logic (Reiter, 1980, 1987), Pollock's Theory of Defeasible Reasoning (Pollock, 1987-1995), Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993), Bondarenko *et al.*'s Assumption-Based Framework for Non-Monotonic Reasoning (Bondarenko *et al.*, 1993), Dung's Argumentation Frameworks (Dung, 1993, 1995), and Loui and Chen's Argument Game (Loui and Chen, 1992).[18]

We do not discuss all argumentation models in full detail, but capture elements that fall within our focus in CumulA. Some acquaintance with the discussed argumentation models is assumed.

6.1 Propositional Logic

We have selected Propositional Logic as an example of an argumentation model without defeat. An argumentation theory capturing elements of Propositional Logic in CumulA can be defined as follows:

Language = L_{PL}, the language of Propositional Logic.
Rules = $\{\{Sentence_1, \ldots, Sentence_n\} \rightarrow Sentence_{n+1} \mid$
$Sentence_1, \ldots, Sentence_n \models_{PL} Sentence_{n+1}\}$,
where $\models_{PL}$ denotes the consequence relation of Propositional Logic.
DefeaterSchemes = $\varnothing$.

The rules of the argumentation theory correspond to logical consequence in Propositional Logic. There are no defeater schemes.

Mostly only single-step arguments are considered, although proof theories for Propositional Logic can be interpreted as descriptions of subordinated arguments from a restricted set of rules. Accounts of Propositional Logic normally do not describe a counterpart of our lines of argumentation. Only maximal sets of conclusions from a set of premises are considered. These are similar to CumulA's forward extensions (restricted to single-step arguments).

This example shows that it is not necessary to explicitly distinguish classes of strict and defeasible arguments, as is done in many argumentation models, e.g. in Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993). If required,

[18] Obvious omissions are the models of Nute (1988), Geffner and Pearl (1992), Simari and Loui (1992), Gordon (1993a, 1993b, 1995), Lodder and Herczog (1995), extending the work of Hage *et al.* (1994), and Prakken and Sartor (1996). All describe significant research, relevant for argumentation, but with a focus different from CumulA's. Nute focuses on a Prolog implementation, Geffner and Pearl on integration of argumentation and the so-called ε-semantics, Simari and Loui on the mathematics of argumentation and specificity, Gordon on dialogue in legal argumentation, Lodder and Herczog on dialogues and commitment, and Prakken and Sartor on defeasible priorities.

an argumentation theory can incorporate a set of arguments that cannot be defeated because the theory does not have defeater schemes that could cause their defeat.[19]

6.2 Poole's Logical Framework for Default Reasoning

We have selected Poole's Logical Framework for Default Reasoning since it is the purest example of consistency maintenance. An argumentation theory capturing elements of Poole's Framework in CumulA can be defined as follows:

Language = L_{PL}, the language of Propositional Logic.
Rules = {{*Sentence*$_1$, ..., *Sentence*$_n$} → *Sentence*$_{n+1}$ |
Sentence$_1$, ..., *Sentence*$_n$ $\models_{PL}$ *Sentence*$_{n+1}$},
where $\models_{PL}$ denotes the consequence relation of Propositional Logic.
DefeaterSchemes = {*Sentence*$_1$, ..., *Sentence*$_{n-1}$ [*Sentence*$_n$] |
Sentence$_1$, ..., *Sentence*$_{n-1}$, *Sentence*$_n$ $\models_{PL}$ $\perp$},
where $\perp$ denotes contradiction in Propositional Logic.

The rules correspond to ordinary logical consequence in Propositional Logic, as in the argumentation theory for Propositional Logic above. The defeater schemes say that an argument is challenged by other arguments if the argument's conclusion is inconsistent with the conclusions of the other arguments.

In Poole's Framework, only single-step arguments are considered. Poole's Framework does not contain a counterpart of our lines of argumentation. Poole's extensions are similar to CumulA's forward extensions.

6.3 Lin and Shoham's Argument Systems

Lin and Shoham's Argument Systems are related to Poole's Logical Framework for Default Reasoning, since both deal mainly with consistency maintenance. We have selected Lin and Shoham's Argument Systems, since in this argumentation model it is recognized that the defeat of arguments can be studied independent of the specific language and argument rules, and that for the study of argument defeat it is useful to consider special sets of structured arguments, such as sets of arguments closed under initials.

An argumentation theory capturing elements of Lin and Shoham's Argument Systems in CumulA can be defined as follows:

Language = *Atoms* ∪ ¬*Atoms*,
where *Atoms* is any set and ¬*Atoms* is the set {¬*Atom* | *Atom* is an element of *Atoms*} (disjoint from *Atoms*).
Rules is any set of rules of the language.

[19] If moreover strict arguments always should defeat defeasible arguments in case of a conflict, additional defeaters are required.

DefeaterSchemes = {$*Atom$ [$*\neg Atom$], $*\neg Atom$ [$*Atom$] | *Atom* is an element of *Atoms*}.

Lin and Shoham abstract from the language used. It is a set of sentences closed under negation. The set of rules is arbitrary. The defeater schemes represent that an argument challenges another if it has opposite conclusion.

Lin and Shoham consider subordinated arguments, and forward lines of argumentation.

6.4 Reiter's Default Logic

Reiter's Default Logic is selected since it rightly remains influential. It should be regarded as an argumentation model avant la lettre. An argumentation theory capturing elements of Reiter's Default Logic in CumulA can be defined as follows:

Language = L_{PL}, the language of Propositional Logic.
Rules $\supseteq$ {{$Sentence_1$, ..., $Sentence_n$} $\rightarrow$ $Sentence_{n+1}$ |
$Sentence_1$, ..., $Sentence_n \models_{PL} Sentence_{n+1}$},
where $\models_{PL}$ denotes the consequence relation of Propositional Logic.
DefeaterSchemes $\subseteq$
{$*\neg$*Justification* [{{$*Condition_1$, ..., $*Condition_n$}} $\rightarrow$ *Conclusion*] |
{$Condition_1$, ..., $Condition_n$} $\rightarrow$ *Conclusion* is an element of *Rules*}.[20]

For convenience, we restricted the language to Propositional Logic. The set of rules is a superset of the set of rules corresponding to ordinary logical consequence. As in Default Logic, rules have so-called justifications. A rule can only be used if its justification is not denied. This leads to defeater schemes of a special form: an argument justifying the negation of a justification of some rule challenges an argument that ends with a step corresponding to the rule. So, a default $Condition_1$, ..., $Condition_n$: $Justification_1$, ..., $Justification_m$ / *Conclusion* of Default Logic corresponds to a rule {$Condition_1$, ..., $Condition_n$} $\rightarrow$ *Conclusion* in *Rules* and defeater schemes $*\neg Justification_i$ [{{$*Condition_1$, ..., $*Condition_n$}} $\rightarrow$ *Conclusion*], for i = 1 to m, in *DefeaterSchemes*. (So, defaults that only differ in their justifications are not distinguished.)

Reiter's Default Logic implicitly describes subordinated arguments and no forward lines of argumentation. Reiter's extensions are similar to CumulA's forward extensions.

---

[20] Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Dung (1995) show how Reiter's (1980, 1987) Default Logic can be translated to their argumentation models. In contrast with us, they also prove formal relations.

6.5 Pollock's Theory of Defeasible Reasoning

Pollock's Theory of Defeasible Reasoning is probably the most worked-out argumentation model. It has been developed and adapted since 1987. An argumentation theory capturing elements of Pollock's theory in CumulA can be defined as follows:

Language = L_{PL}, the language of Propositional Logic.
Rules $\supseteq$ {{*Sentence*$_1$, ..., *Sentence*$_n$} $\rightarrow$ *Sentence*$_{n+1}$ |
Sentence$_1$, ..., *Sentence*$_n$ $\models_{PL}$ *Sentence*$_{n+1}$},
where $\models_{PL}$ denotes the consequence relation of Propositional Logic.
CollectiveDefeat $\subseteq$ *DefeaterSchemes* $\subseteq$ *Undercutters* $\cup$ *Rebutters* $\cup$ *CollectiveDefeat*,
where
CollectiveDefeat =
{[{*Subreason*$_{11}$, ..., *Subreason*$_{n1}$} $\rightarrow$ *Conclusion*$_1$, ...,
{*Subreason*$_{1m}$, ..., *Subreason*$_{nm}$} $\rightarrow$ *Conclusion*$_m$] |
Conclusion$_1$, ..., *Conclusion*$_n$ is minimally inconsistent},
Undercutters =
{*Conclusion*$_1$ [{*Subreason*$_1$, ..., *Subreason*$_n$} $\rightarrow$ *Conclusion*$_2$]},
and
Rebutters =
{{*Subreason*$_{11}$, ..., *Subreason*$_{n1}$} $\rightarrow$ *Conclusion*$_1$
[{*Subreason*$_{12}$, ..., *Subreason*$_{n2}$} $\rightarrow$ *Conclusion*$_2$] |
Conclusion$_1$, *Conclusion*$_2$ $\models_{PL}$ $\perp$}.

Again, the set of rules is a superset of the rules corresponding to ordinary logical consequence. The defeater schemes are of three forms: those representing collective defeat (restricted to arguments with inconsistent conclusions), undercutting defeat, and rebutting defeat (see chapter 5, section 3.5, 3.1, and 3.2, respectively). Since Pollock uses collective defeat as a general means to preserve consistency, the set of defeater schemes is a superset of the set of defeater schemes representing collective defeat.

Pollock describes subordinated arguments and forward lines of argumentation.

6.6 Vreeswijk's Abstract Argumentation Systems

Vreeswijk's Abstract Argumentation Systems have been selected since Vreeswijk's argumentation model has influenced the development of CumulA (see chapter 5). Vreeswijk's model can be regarded as a refinement of Lin and Shoham's Argument Systems. An argumentation theory capturing elements of Vreeswijk's Abstract Argumentation Systems in CumulA can be defined as follows:

Language is any set, containing ⊥, denoting contradiction.
Rules is any set of rules in the language.
DefeaterSchemes ⊆ {*Argument*$_1$, ..., *Argument*$_{n-1}$ [*Argument*$_n$] |
There is a rule {Conclusion(*Argument*$_1$), ..., Conclusion(*Argument*$_n$)} → ⊥}.

Just as Lin and Shoham's Argument Systems and CumulA, Vreeswijk's model is independent of a specific language; Vreeswijk's language only contains a special element denoting contradiction. The set of rules is arbitrary. The defeater schemes of Vreeswijk's model represent that an argument is challenged by other arguments, if the argument's conclusion is inconsistent with the conclusions of the other arguments. The defeater schemes resemble those of the theory capturing elements of Lin and Shoham's Argument Systems. However, there are three differences. First, Vreeswijk notion of inconsistency is somewhat more general than Lin and Shoham's since it includes inconsistency of more than two arguments. Second, only a subset of the defeater schemes is used. Which defeater schemes are selected depends on Vreeswijk's conclusive force relation, included in each Abstract Argumentation System, in the following way: for arguments *Argument*$_1$, ..., *Argument*$_n$, such that there is a rule {Conclusion(*Argument*$_1$), ..., Conclusion(*Argument*$_n$)} → ⊥, the fact that for some i, $1 \leq i \leq n$, *Argument*$_i$ has less conclusive force than *Argument*$_n$ implies that *Argument*$_1$, ..., *Argument*$_{n-1}$ [*Argument*$_n$] is not in *DefeaterSchemes*.[21] Third, the defeater schemes corresponding to Vreeswijk's model are of composite-type, whereas those of Lin and Shoham's model are of sentence-type. This is the result of the fact that Vreeswijk's conclusive force relation is a relation between full arguments.

Vreeswijk's model describes subordinated arguments and forward lines of argumentation.

6.7 Bondarenko *et al.*'s Assumption-Based Framework

Bondarenko *et al.*'s Assumption-Based Framework for Non-Monotonic Reasoning have been selected since the formalism has a specific type of defeat, that is worth distinguishing: assumption-type defeat. An argumentation theory capturing this specific element of Bondarenko *et al.*'s Assumption-Based Framework in CumulA can be defined as follows:

Language = *Atoms* ∪ ¬*Atoms*,
where *Atoms* is any set and ¬*Atoms* is the set {¬*Atom* | *Atom* is an element of *Atoms*} (disjoint from *Atoms*).
Rules is any set of rules in the language.
DefeaterSchemes ⊆ {**Atom* [¬*Atom*], *¬*Atom* [*Atom*] | *Atom* is an element of *Atoms*}.

[21] It could be interesting to establish formal connections between properties of a conclusive force relation and those of the corresponding set of defeater schemes.

This theory is related to the one capturing elements of Lin and Shoham's Argument Systems. The language is a set closed under negation and the set of rules is arbitrary. However, the defeater schemes differ subtly from those in the theory capturing elements of Lin and Shoham's, in two ways. First, the challenged arguments in the instances of the defeater schemes are statements. As a result, argument defeat of a non-statement argument is always indirect (see chapter 5, sections 4.3 and 4.4), because of the defeat of a premise of the argument. The defeater schemes are of assumption-type (see section 2). Second, not all defeater schemes of the given form need to be included in the argumentation theory. If $*Atom$ [$\neg Atom$] (or $*\neg Atom$ [$Atom$]) is included, $\neg Atom$ (or $Atom$, respectively) is called an assumption of the theory. Intuitively, an assumption can be the premise of an undefeated argument, unless its negation is justified.

Bondarenko *et al.*'s model implicitly describes subordinated arguments and forward lines of argumentation.

6.8 Dung's Argumentation Frameworks

Dung's Argumentation Frameworks have been selected since Dung has brought the abstract study of argumentation and defeat to its extreme. Dung notices that the basis of defeat is the attack relation between arguments. As a result, he focuses on that relation, independent of the structure of the arguments involved. This is an important step towards a better understanding of argumentation and defeat.

An argumentation theory capturing elements of Dung's Argumentation Frameworks in CumulA can be defined as follows:

Language is any set.
Rules = ∅.
DefeaterSchemes ⊆ {*Statement*$_1$ [*Statement*$_2$]}

As Lin and Shoham's Argument Systems, Vreeswijk's Abstract Argumentation Systems and CumulA, Dung's model is independent of a specific language. Moreover, Dung abstracts from the structure of arguments. As a result, the set of rules is empty. The defeater schemes - actually defeaters - are all simple defeaters.

Dung considers unstructured arguments, corresponding to CumulA's statements, and no lines of argumentation. Verheij (1996a) investigates the formal relations between Dung's model and the stages approach of CumulA.

6.9 Loui and Chen's Argument Game

Loui and Chen's Argument Game has been selected since it shows a characteristic of argumentation not found in any of the other discussed argumentation models: backward argumentation. The Argument Game is a two-player card game, designed as a model of argumentation. One of the players tries to justify a conclusion by means of an undefeated argument, the other tries to challenge the

argument. As a result, the conclusion is fixed, while the premises vary throughout the game.

An argumentation theory capturing elements of Loui and Chen's Argument Game in CumulA can be defined as follows:

Language = *Atoms* $\cup$ $\neg$*Atoms*,
where *Atoms* is any set and $\neg$*Atoms* is the set {$\neg$*Atom* | *Atom* is an element of *Atoms*} (disjoint from *Atoms*).
Rules is any set of rules in the language.
DefeaterSchemes $\subseteq$ {**Atom* [*$\neg$*Atom*], *$\neg$*Atom* [**Atom*] | *Atom* is an element of *Atoms*}

Surprisingly, this argumentation theory is the same as the one capturing elements of Lin and Shoham's Argument Systems. This shows that the underlying notions of argument and defeat are the same in both models. However, argumentation is different in both models, since Loui and Chen consider backward lines of argumentation. Moreover, other differences between the models have disappeared, since we only focus on the underlying model of argumentation, and have therefore abstracted from the game elements of the Argument Game, such as bidding and the different roles of the players.

The arguments of Loui and Chen's Argument Game are constructed by subordination. The game models backward lines of argumentation with a single fixed conclusion.

7 A comparison of argumentation models

After capturing elements of several argumentation models as argumentation theories in CumulA in the previous section, we now apply the distinctions discussed in the sections 1 to 5 to those argumentation theories. An overview is given in table 1. The table shows differences and similarities.

We have shown the generality of CumulA by capturing elements of selected argumentation models in CumulA. Previously, Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Dung (1995) have captured other selections of argumentation models in their formalisms. We stress that, in contrast with us, they have also proven formal relations.

Lin (1993) has also classified formalisms of nonmonotonic reasoning, using a distinction based on intuition. He distinguished two classes, namely sentence-based and argument-based formalisms. His distinction seems to be close to our distinction of sentence-type and composite-type defeat. Interestingly, in a footnote, Lin (1993, note 1, p. 254) remarks that Default Logic (Reiter, 1980) should probably be classified in both categories. We are able to clarify the position of Default Logic by classifying it in the intermediate class of step-type defeat.

Argumentation model	Type of arguments	Argument structure and defeat	Individual or group-wise defeat	Trigger of defeat	Direction of argumentation
Propositional Logic	single-step or subordination	no defeat	no defeat	no defeat	static
Poole's Logical Framework for Default Reasoning	single-step	sentence-type defeat	self-defeat and left-compound defeat	inconsistency-triggered defeat	static
Lin and Shoham's Argument Systems	subordination	sentence-type defeat	simple defeat	inconsistency-triggered defeat	forward argumentation
Reiter's Default Logic	subordination	step-type defeat	simple defeat	counterargument-triggered defeat	static
Pollock's Theory of Defeasible Reasoning	subordination	sentence-type and step-type defeat	self-defeat, simple and right-compound defeat	counterargument-triggered defeat	forward argumentation
Vreeswijk's Abstract Argumentation Systems	subordination	composite-type defeat	self-defeat and left-compound defeat	inconsistency-triggered defeat	forward argumentation
Bondarenko *et al.*'s Assumption-Based Framework	subordination	assumption-type defeat	simple defeat	inconsistency-triggered defeat	forward argumentation
Dung's Argumentation Frameworks	statements	sentence-type defeat	simple defeat	counterargument-triggered defeat	static
Loui and Chen's Argument Game	subordination	sentence-type defeat	simple defeat	inconsistency-triggered defeat	backward argumentation
CumulA	subordination and coordination	all types	all types	counterargument-triggered defeat	bidirectional argumentation

Table 1: A comparison of argumentation models

Since we focused on the argumentation theories capturing elements of argumentation models in CumulA, we were able to establish a number of distinctions on formal grounds in contrast with Lin's distinction based on intuition. As a result, we have shown similarities and differences between the argumentation models.

Chapter 7

Results and conclusions

In chapter 1, sextion 7, we discussed the research questions and goals of the thesis. In this final chapter, we summarize the results and conclusions. We do this in three parts: rules and reasons (section 1), legal reasoning (section 2), and dialectical argumentation (section 3). We close with some suggestions for future research (section 4).

1 Rules and reasons

Our first group of research questions (chapter 1, section 7) was the following:

- What is the role of rules and reasons in argumentation with defeasible arguments? What properties of rules and reasons are relevant for argumentation and defeat? How do these properties relate?

In order to answer these questions, we have presented a formal model of rules and reasons as they are used in argumentation: Reason-Based Logic. The formalism is a formal semantics of rules and reasons; it focuses on the types of facts relevant for argumentation with defeasible arguments, and the relations between these facts.

We established the following types of facts concerning rules and reasons that are relevant for argumentation with defeasible arguments:

- The state of affairs *state-of-affairs*$_1$ is a reason for the state of affairs *state-of-affairs*$_2$.
- There is a valid rule with condition *condition* and conclusion *conclusion*.
- The rule with condition *condition* and conclusion *conclusion* is excluded for the instance *fact* of its condition.
- The rule with condition *condition* and conclusion *conclusion* is made applicable by the fact expressed by the instance *fact* of its condition.
- The rule with condition *condition* and conclusion *conclusion* applies on the basis the fact expressed by the instance *fact* of its condition.
- The reasons *reasons-pro* for the conclusion *conclusion* outweigh thr reasons *reasons-con* against it.

In chapter 2, the relations between these types of facts are elaborated in the formalism Reason-Based Logic.

In Reason-Based Logic, there are three main mechanisms that lead to defeat:

1. An exclusionary reason makes a rule inapplicable (cf. Raz, 1990).
2. Reasons for a conclusion do not lead to that conclusion if the reasons against the conclusion outweigh the reasons for it (cf. Naess, 1978).
3. A rule does not apply if the reasons against applying the rule outweigh the reasons for applying it.

In chapter 2, these are worked out in detail. The use of exclusionary reasons is closely related to the use of exception predicates, well-known in the research on nonmonotonic reasoning (cf., e.g., Prakken, 1993). Although there are several formalisms that model some form of weighing of reasons, Reason-Based Logic is, as far as we know, the first in which weighing is treated qualitatively instead of quantitatively. We know of no other formalism that models reasons for and against applying a rule.

Once again we stress that there is no single generally agreed upon interpretation of the notions 'rule' and 'reason'. As the many versions of Reason-Based Logic[1] show, this is not even the case if one restricts oneself to the rules and reasons of argumentation with defeasible arguments.

Therefore our formalism is accompanied by many examples in order to make the interpretation of the notions rule and reason as clear as possible (cf. our method of research, described in chapter 1, section 7).

Apart from the particular form of Reason-Based Logic as presented in this thesis, we have made three general contributions to the research on the formalization of rules and reasons:

1. We have separated the semantics of rules and reasons, as used in argumentation with defeasible arguments, from the definition of a defeasible consequence relation. Although this is similar to the preferential-model semantics for nonmonotonic consequence relations (Shoham, 1988; Kraus *et al.*, 1990; Makinson, 1994), there is a difference: in Reason-Based Logic, the facts concerning rules and reasons related to defeat are explicitly represented in the logical language, while the preference relation (that determines nonmonotonicity) of a preferential-model semantics is separated from the logical language. In this way, the definition of defeasible reasoning in Reason-Based Logic becomes less ad hoc, and is based on explicit standards (cf. chapter 2, section 6).
2. We have shown that it is advantageous to consider rules as special objects and to use a translation from sentences to terms (cf. chapter 2, section 4). In this

[1] E.g., Hage (1991, 1993, 1995), Hage and Verheij (1994a, b), Hage *et al.* (1993), Verheij (1994, 1995c), Verheij and Hage (1994).

way, it becomes possible to represent facts about rules, and to reason with them. As a result, we could keep the merits of two competing approaches: the use of rule identifiers and the use of special-purpose conditionals. Our approach enhances the ad hoc use of rule identifiers, which was introduced in order to represent facts about rules. At the same time, our approach can represent the validity of rules, which is an advantage of the use of special-purpose conditionals in contrast with the use of rule identifiers (cf. chapter 4).

3. We have separated the generation of a reason and the generation of a conclusion, which can both occur when the condition of a rule is satisfied. First, this clarifies the relation of rules and reasons, and second, this allows different levels where defeasibility can occur (cf. chapter 3, sections 5 and 6).

2 Legal reasoning

Legal reasoning has been an important inspiration during the development of Reason-Based Logic. Legal reasoning provides good examples for Reason-Based Logic, since in the law several pragmatic solutions have been developed to dealing with exceptions to rules, dealing with rule conflicts, and reasoning about rules. As a result, the usefulness of Reason-Based Logic can be shown using examples from the field of law.

In chapter 3, we have formalized several examples of legal reasoning in Reason-Based Logic. Apart from different ways of dealing with exceptions to rules and rule conflicts, which are specific for Reason-Based Logic, we have given two applications of Reason-Based Logic to the theory of legal reasoning, namely to integrating rules and principles, and to reasoning by analogy:

1. We have presented an integrated view on rules and principles, and have shown that rules and principles can be regarded as the extremes of a spectrum of hybrid rules/principles. This integrated view is in contrast with Dworkin's strict distinction between rules and principles (cf. Dworkin, 1978).
2. We have given three different ways of reconstructing reasoning by analogy: (1) application of principles that underlie the original rule, (2) application of an analogous rule/principle that has the same underlying principles as the original rule, and (3) analogous application of the original rule, i.e., the application of the rule with non-standard justification. The first of these ways of reconstruction of reasoning by analogy follows directly from the integrated view on rules and principles. The second is a familiar interpretation of analogy, except that we have made the nature and justification of the analogy explicit in terms of underlying principles. The third is typical for Reason-Based Logic.

Since we have given formal elaborations in Reason-Based Logic, the insights can be applied to the use of computers as tools in the field of law (cf. Van den Herik, 1991).

3 Dialectical argumentation

The second group of research questions (chapter 1, section 7) was the following:

- What is the role of process in argumentation with defeasible arguments? How is the defeat of an argument determined by its structure, counterarguments and the argumentation stage?

In order to answer these questions, we developed a formal model of dialectical argumentation, CumulA, in chapter 5.

We have focused on the process of taking arguments into account, and on the defeasibility of arguments. CumulA is a model in which the defeat status of an argument, either undefeated or defeated, depends on:

1. the structure of the argument;
2. counterarguments;
3. the argumentation stage.

We discuss each below.

In CumulA, the structure of arguments is modeled as in, e.g., the argumentation theory of Van Eemeren *et al.* (1981, 1987). Both the subordination and the coordination of arguments are possible. In CumulA, it is explored how the structure of arguments can lead to their defeat. To our knowledge, CumulA is the only formalism that explores how the coordination of arguments influences defeat (cf. the definitions of the narrowings of arguments in chapter 5, section 2.4, and of defeat status assignments in chapter 5, section 4.4).

In CumulA, the influence of counterarguments on defeat is modeled using defeaters. Defeaters indicate when arguments can defeat other arguments. We have shown that defeaters can be used to represent a wide range of types of defeat: undercutting and rebutting defeat, as distinguished by, e.g., Pollock (1987), defeat by sequential weakening and by parallel strengthening, as distinguished by Verheij (1995c), and collective and indeterministic defeat, related to the well-known skeptical and credulous approaches in nonmonotonic reasoning (cf. Ginsberg, 1987). However, these types of defeat were not previously integrated in one formalism (cf. chapter 5, section 3).

Argumentation stages represent the arguments taken into account and the status of these arguments, either defeated or undefeated. CumulA's lines of argumentation, formally sequences of stages, give insight into the influence that the process of taking arguments into account has on the status of arguments. For instance, by means of argumentation diagrams, which give an overview of possible lines of argumentation, phenomena that are characteristic for argumentation with defeasible arguments, such as the reinstatement of arguments, are explicitly depicted.

In chapter 6, we have analyzed a number of existing argumentation models. First, we made several formal distinctions between argumentation theories.

- Four types of arguments were distinguished in CumulA by their structure: statements, single-step arguments, arguments that are constructed by subordination, and arguments that are constructed by subordination and coordination.
- Four types of defeat were distinguished by the structure of the challenging and challenged arguments involved: no defeat, sentence-type defeat (with, as a special case, assumption-type defeat), step-type defeat, and composite-type defeat.
- Five types of defeat were distinguished by the number of challenging and challenged arguments involved: no defeat, self-defeat, simple defeat, left-compound defeat, and right-compound defeat.
- Two types of defeat were distinguished by different ways in which defeat is triggered: inconsistency-triggered and counterargument-triggered defeat.
- Four types of direction of argumentation were distinguished: static argumentation, forward argumentation, backward argumentation, and bi-directional argumentation.

Second, we have shown the generality of CumulA by capturing elements of selected argumentation models in CumulA. Previously, Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Dung (1995) have captured other selections of argumentation models in their formalisms. However, we have not proven formal relations, in contrast with Lin and Shoham and Dung.

Third, we have shown similarities and differences between the argumentation theories capturing argumentation models by applying the distinctions above. Previously, Lin (1993) made a distinction related to our distinction of sentence-type and composite-type defeat. However, his distinction was based on intuition, while ours is based on formal grounds. Moreover, we have made several other distinctions.

To conclude, CumulA has shown that

1. it is advantageous to consider arguments structured both by subordination and by coordination if argumentation with defeasible arguments is modeled;
2. the defeat of arguments can be described in terms of their structure, counterarguments, and the stage of the argumentation process;
3. both forward and backward argumentation can be formalized in one model.

4 Future research

A first direction of future research will be the integratation of the ideas behind Reason-Based Logic and CumulA. Whereas Reason-Based Logic lacks a process

model of argumentation, like that of CumulA, CumulA lacks a rich language, like that of Reason-Based Logic. Because of the already existing connections between the two models, e.g., the admission of the accrual of reasons, this direction of research could be fruitful, and could lead to a better understanding of argumentation with defeasible arguments.

A second direction of future research will be the implementation of Reason-Based Logic and CumulA. Early versions of Reason-Based Logic have been implemented in Prolog (Hage, 1993; Verheij, 1993, 1995e), but have become outdated by the later theoretical enhancements. CumulA has not been implemented, but seems to be well-suited, due to its process-orientation. Moreover, it is promising that Dung (1995) has shown close connections between argumentation and logic programming.

A third direction of future research will be the practical assessment of the mostly theoretically motivated ideas on legal reasoning, as presented in this thesis. Probably, the actual legal practice will necessitate several adjustments and compromises. There is a detailed plan to test the theoretical ideas against the actual practice in the legal domain of tort.[2]

---

[2] The Dutch National Program for Information Technology and Law (ITeR) has recently provided funding for this project, that will be carried out at the Department of Metajuridica of the Universiteit Maastricht.

References

Ashley, K. (1990). *Modeling legal argument. Reasoning with cases and hypotheticals.* The MIT Press, Cambridge (Massachusetts).

Bench-Capon, T.J.M. (1995). Argument in Artificial Intelligence and Law. *Legal knowledge based systems. Telecommunication and AI & Law* (eds. J.C. Hage, T.J.M. Bench-Capon, M.J. Cohen and H.J. van den Herik), pp. 5-14. Koninklijke Vermande, Lelystad.

Bench-Capon, T.J.M., and Coenen, F.P. (1992). Isomorphism and legal knowledge based systems. *Artificial Intelligence and Law*, Vol. 1, pp. 65-86.

Bondarenko, A., Toni, F., and Kowalski, R.A. (1993). An assumption-based framework for non-monotonic reasoning. *Logic programming and non-monotonic reasoning. Proceedings of the second international workshop* (eds. L.M. Pereira and A. Nerode), pp. 171-189. The MIT Press, Cambridge (Massachusetts).

Brewka, G. (1994). A Reconstruction of Rescher's Theory of Formal Disputation Based on Default Logic. *ECAI 94. 11th European Conference on Artificial Intelligence* (ed. A.G. Cohn), pp. 366-370. John Wiley & Sons, Chichester.

Copi, I.M. (1982). *Introduction to logic. Sixth edition.* Macmillan Publishing Co., New York (New York).

Dalen, D. van (1983). *Logic and Structure. Second Edition.* Springer-Verlag, Berlin.

Davis, E. (1990). *Representations of Commonsense Knowledge.* Morgan Kaufmann Publishers, San Mateo (California).

Davis, M. (1993). First order logic. *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 1. Logical Foundations* (eds. D.M. Gabbay, C.J. Hogger and J.A. Robinson), pp. 31-65. Clarendon Press, Oxford.

Dawkins, R. (1989). *The Selfish Gene. New Edition.* Oxford University Press, Oxford.

Delgrande, J. (1988). An approach to default reasoning based on a first-order conditional logic: revised report. *Artificial Intelligence*, Vol. 36, pp. 63-90.

Dung, P.M. (1993). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning and logic programming. *IJCAI-93. Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (ed. Ruzena Bajcsy), pp. 852-857. Morgan Kaufmann Publishers, San Mateo (California).

Dung, P.M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, Vol. 77, pp. 321-357.

Dworkin, R. (1978). *Taking Rights Seriously. New Impression with a Reply to Critics.* Duckworth, London.

Eemeren, F.H. van, Grootendorst, R., and Kruiger, T. (1981). *Argumentatietheorie.* Uitgeverij Het Spectrum, Utrecht.

Eemeren, F.H. van, Grootendorst, R. and Kruiger, T. (1987). *Handbook of Argumentation Theory. A Critical Survey of Classical Backgrounds and Modern Studies.* Foris Publications, Dordrecht. Translation of Van Eemeren *et al.* (1981).

Fuller, L.L. (1958). Positivism and fidelity to law: A reply to Professor Hart. *Harvard Law Review*, Vol. 71, pp. 630-672.

Gabbay, D.M., Hogger, C.J., and Robinson, J.A. (eds.) (1993). *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 1. Logical Foundations.* Clarendon Press, Oxford.

Gabbay, D.M., Hogger, C.J., and Robinson, J.A. (eds.) (1994a). *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 2. Deduction Methodologies.* Clarendon Press, Oxford.

Gabbay, D.M., Hogger, C.J., and Robinson, J.A. (eds.) (1994b). *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 3. Nonmonotonic Reasoning and Uncertain Reasoning.* Clarendon Press, Oxford.

Gabbay, D.M., and Ohlbach, H.J. (eds.) (1996). *Practical Reasoning. International Conference on Formal and Applied Practical Reasoning (FAPR '96; Lecture Notes in Artificial Intelligence, Vol. 1085).* Springer-Verlag, Berlin.

Geffner, H., and Pearl, J. (1992). Conditional entailment: bridging two approaches to default reasoning. *Artificial Intelligence*, Vol. 53, pp. 209-244.

Ginsberg, M.L. (ed.) (1987). *Readings in Nonmonotonic Reasoning.* Morgan Kaufmann Publishers, Los Altos (California).

Gordon, T.F. (1993a). *The Pleadings Game. An Artificial Intelligence Model of Procedural Justice.* Dissertation.

Gordon, T.F. (1993b). The Pleadings Game. Formalizing procedural justice. *Fourth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 10-19. ACM, New York (New York).

Gordon, T.F. (1995). *The Pleadings Game. An Artificial Intelligence Model of Procedural Justice.* Kluwer Academic Publishers, Dordrecht.

Haack, S. (1978). *Philosophy of logics.* Cambridge University Press, Cambridge.

Hage, J.C. (1991). Monological reason based reasoning. *Legal knowledge based systems. Model-based legal reasoning* (eds. J.A. Breuker, R.V. de Mulder and J.C. Hage), pp. 77-91. Vermande, Lelystad.

Hage, J.C. (1993). Monological reason based logic. A low level integration of rule-based reasoning and case-based reasoning. *The Fourth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 30-39. ACM, New York (New York). Also published as report SKBS/B3.A/93-08.

Hage, J.C. (1995). Teleological reasoning in Reason-Based Logic. *The Fifth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 11-20. ACM, New York (New York).

Hage, J.C., Leenes, R., and Lodder, A.R. (1994). Hard Cases: A Procedural Approach. *Artificial Intelligence and Law*, Vol. 2, pp. 113-167.

Hage, J.C. and Verheij, B. (1994a). Reason-based Logic: a logic for reasoning with rules and reasons. *Law, Computers and Artificial Intelligence*, Vol. 3, No. 2/3, pp. 171-209. Also published as report SKBS/B3.A/94-10.

Hage, J.C., and Verheij, B. (1994b). Towards a logic for reasoning with norms. *ECAI'94 Workshop W9, Artificial Normative Reasoning* (ed. J. Breuker), pp. 160-177. Also published as report SKBS/B3.A/94-12.

Hage, J.C., Verheij, B., and Lodder, A.R. (1993). Reason based logic. A logic that deals with rules and reasons. *Working Papers NAIC '93* (eds. J.M. Akkermans and J.A. Breuker), pp. 293-304. Also published as report SKBS/B3.A/93-19.

Hayes, P.J. (1985). The Second Naive Physics Manifesto. *Formal Theories of the Commonsense World* (eds. J.R. Hobbs and R.C. Moore), pp. 1-36. Ablex Publishing Corporation, Norwood (New Jersey).

Herik, H.J. van den (1991). *Kunnen computers* rechts*preken?* Gouda Quint, Arnhem.

Hobbs, J.R., and Moore, R.C. (eds.) (1985). *Formal Theories of the Commonsense World.* Ablex Publishing Corporation, Norwood (New Jersey).

Kraus, S., Lehmann, D., and Magidor, M. (1990). Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, Vol. 44, pp. 167-207.

Leenes, R.E., Lodder, A.R., and Hage, J.C. (1994). A Dialogue Game for Legal Arguments. *Law, Computers & Artificial Intelligence*, Vol. 3, No. 2/3, pp. 112-122. Also published as report SKBS/B3.A/94-06.

Lin, F., and Shoham, Y. (1989). Argument Systems: a uniform basis for nonmonotonic reasoning. *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning* (eds. R.J. Brachman, H.J. Levesque and R. Reiter), pp. 245-255. Morgan Kaufmann Publishers, San Mateo (California).

Lin, F. (1993). An argument-based approach to nonmonotonic reasoning. *Computational Intelligence*, Vol. 9, No. 3, pp. 254-267.

Lodder, A.R., and Herczog, A. (1995). DiaLaw. A dialogical framework for modeling legal reasoning. *The Fifth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 146-155. ACM, New York (New York).

Lodder, A.R. (1996). The ideas behind DiaLaw, a procedural model for legal justification. *Proceedings of the Fifth National/First European Conference on Law, Computers and Artificial Intelligence* (eds. I. Carr and A. Narayanan), pp. 93-103. Exeter, England. An abstract is available on the World-Wide Web at http://www.metajur.unimaas.nl/~arno/exeter96.htm.

Loui, R.P. (1987). Defeat among arguments: a system of defeasible inference. *Computational Intelligence*, Vol. 3, No. 2, pp. 100-106.

Loui, R.P. (1991). Argument and belief: Where we stand in the Keynesian tradition. *Minds and machines,* Vol. 1, pp. 357-365.

Loui, R.P. (1992). Process and Policy: Resource-Bounded Non-Demonstrative Reasoning. *Report WUCS-92-43.* Washington University, Department of Computer Science, Saint Louis (Missouri).

Loui, R.P. (1995a). Hart's critics on defeasible concepts and ascriptivism. *The Fifth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 21-30. ACM, New York (New York).

Loui, R.P. (1995b). The Workshop on Computational Dialectics. *AI Magazine*, Vol. 16, No. 4, pp. 101-104.

Loui, R.P., and Chen, W. (1992). An Argument Game. *Report WUCS-92-47.* Department of Computer Science, Washington University, Saint Louis (Missouri).

Lukaszewicz, W. (1990). *Non-monotonic reasoning. Formalization of commonsense reasoning.* Ellis Horwood, New York (New York).

Makinson, D. (1994). General Patterns in Nonmonotonic Reasoning. *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 3. Nonmonotonic Reasoning and Uncertain Reasoning* (eds. D.M. Gabbay, C.J. Hogger and J.A. Robinson), pp. 35-110. Clarendon Press, Oxford.

Naess, A. (1978). *Elementaire argumentatieleer*. Ambo, Baarn.

Nute, D. (1980). *Topics in conditional logic*. D. Reidel Publishing Company, Dordrecht.

Nute, D. (1988). Defeasible reasoning: a philosophical analysis in Prolog. *Aspects of Artificial Intelligence* (ed. James H. Fetzer), pp. 251-288. Kluwer Academic Publishers, Dordrecht.

Nute, D. (1994). Defeasible Logic. *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 3. Nonmonotonic Reasoning and Uncertain Reasoning* (eds. D.M. Gabbay, C.J. Hogger and J.A. Robinson), pp. 353-395. Clarendon Press, Oxford.

Perlis, D. and Subrahmanian, V.S. (1994). Meta-languages, Reflection Principles and Self-Reference. *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 2. Deduction Methodologies* (eds. D.M. Gabbay, C.J. Hogger and J.A. Robinson), pp. 323-358. Clarendon Press, Oxford.

Pollock, J.L. (1986). *Contemporary Theories of Knowledge*. Rowman & Littlefield, Totowa (New Jersey).

Pollock, J.L. (1987). Defeasible reasoning. *Cognitive Science*, Vol. 11, pp. 481-518.

Pollock, J.L. (1991a). A theory of defeasible reasoning. *International Journal of Intelligent Systems*, Vol. 6, pp. 33-54.

Pollock, J.L. (1991b). Self-defeating arguments. *Minds and Machines*, Vol. 1, pp. 367-392.

Pollock, J.L. (1992). How to reason defeasibly. *Artificial Intelligence*, Vol. 57, pp. 1-42.

Pollock, J.L. (1994). Justification and defeat. *Artificial Intelligence*, Vol. 67, pp. 377-407.

Pollock, J.L. (1995). *Cognitive Carpentry: A Blueprint for How to Build a Person.* The MIT Press, Cambridge (Massachusetts).

Poole, D. (1988). A logical framework for default reasoning. *Artificial Intelligence*, Vol. 36, pp. 27-47.

Prakken, H. (1993a). *Logical tools for modelling legal argument.* Doctoral thesis, Vrije Universiteit, Amsterdam.

Prakken, H. (1993b). A logical framework for modelling legal argument. *The Fourth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 1-9. ACM, New York (New York).

Prakken, H. (1995). A semantic view on reasoning about priorities (extended abstract). *Dutch/German Workshop on Nonmonotonic Reasoning. Proceedings of the Second Workshop*, pp. 152-159. Delft University of Technology, Universiteit Utrecht.

Prakken, H., and Sartor, G. (1996). A system for defeasible argumentation, with defeasible priorities. *Practical Reasoning. International Conference on Formal and Applied Practical Reasoning (FAPR '96; Lecture Notes in Artificial Intelligence, Vol. 1085)* (eds. D.M. Gabbay and H.J. Ohlbach), pp. 510-524. Springer-Verlag, Berlin.

Purtill, R.L. (1979). *Logic. Argument, refutation, and proof.* Harper & Row, New York (New York).

Raz, J. (1990). *Practical Reason and Norms.* Princeton University Press, Princeton (New Jersey).

Read, S. (1995). *Thinking About Logic. An Introduction to the Philosophy of Logic.* Oxford University Press, Oxford.

Reiter, R. (1980). A Logic for Default Reasoning. *Artificial Intelligence*, Vol. 13, pp. 81-132.

Reiter, R. (1987). A Logic for Default Reasoning. *Readings in Nonmonotonic Reasoning* (ed. M.L. Ginsberg), pp. 68-93. Morgan Kaufmann Publishers, Los Altos (California). Reprinted from Reiter (1980).

Reiter, R., and Criscuolo, G. (1981). On interacting defaults. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence (IJCAI 81)*, pp. 270-276. Morgan Kaufmann Publishers, Los Altos (California).

Reiter, R., and Criscuolo, G. (1987). On interacting defaults. *Readings in Nonmonotonic Reasoning* (ed. M.L. Ginsberg), pp. 94-100. Morgan Kaufmann Publishers, Los Altos (California). Reprinted from Reiter and Criscuolo (1981).

Rescher, N. (1977). *Dialectics. A controversy-oriented approach to the theory of knowledge.* State University of New York Press, Albany (New York).

Rescher, N. (1988). *Rationality. A Philosophical Inquiry into the Nature and the Rationale of Reason.* Clarendon Press, Oxford.

Sanford, D.H. (1989). *If P, then Q. Conditionals and the Foundations of Reasoning.* Routledge, London.

Sartor, G. (1994). A formal model of legal argumentation. *Ratio Juris*, Vol. 7, No. 2, pp. 177-211.

Shoham, Y. (1988). *Reasoning about change. Time and causation from the standpoint of artificial intelligence.* The MIT Press, Cambridge (Massachusetts).

Simari, G.R., and Loui, R.P. (1992). A mathematical treatment of defeasible reasoning and its applications. *Artificial Intelligence*, Vol. 53, pp. 125-157.

Soeteman, A. (1991). Hercules aan het werk. Over de rol van rechtsbeginselen in het recht. *Rechtsbeginselen*, pp. 41-56. Ars Aequi, Nijmegen.

Stewart, I. (1996). *From here to infinity. A guide to today's mathematics.* Oxford University Press, Oxford.

Tiscornia, D. (1994). Three meanings of analogical reasoning in law. *Proceedings of the Fourth National Conference on Law, Computers and Artificial Intelligence* (eds. I. Carr and A. Narayanan), pp. 137-153. University of Exeter.

Toulmin, S.E. (1958). *The uses of argument.* University Press, Cambridge.

Verheij, B. (1993). *Reason Based Logic in Law.* Report SKBS/B3.A/93-18. Also published as Verheij (1995d).

Verheij, B. (1994). Reason Based Logic and legal knowledge representation. *Proceedings of the Fourth National Conference on Law, Computers and Artificial Intelligence* (eds. I. Carr and A. Narayanan), pp. 154-165. University of Exeter. Also published as report SKBS/B3.A/94-05.

Verheij, B. (1995a). Accrual of arguments in defeasible argumentation. *Dutch/German Workshop on Nonmonotonic Reasoning. Proceedings of the Second Workshop*, pp. 217-224. Delft University of Technology, Universiteit Utrecht. Also published as report SKBS/B3.A/95-01.

Verheij, B. (1995b). The influence of defeated arguments in defeasible argumentation. *WOCFAI 95. Proceedings of the Second World Conference on the Fundamentals of Artificial Intelligence* (eds. M. De Glas and Z. Pawlak), pp. 429-440. Angkor, Paris. Also published as report SKBS/B3.A/95-03. An abstract is available on the World-Wide Web at http://www.cs.unimaas.nl/~verheij/papers/wocfai95.htm.

Verheij, B. (1995c). Arguments and defeat in argument-based nonmonotonic reasoning. *Progress in Artificial Intelligence. 7th Portuguese Conference on Artificial Intelligence (EPIA '95; Lecture Notes in Artificial Intelligence 990)* (eds. Carlos Pinto-Ferreira and Nuno J. Mamede), pp. 213-224. Springer-Verlag, Berlin. Also published as report SKBS/B3.A/95-04. An abstract is available on the World-Wide Web at http://www.cs.unimaas.nl/~verheij/papers/epia95.htm.

Verheij, B. (1995d). Reason Based Logic in Law. *Verso un sistema esperto giuridico integrale* (eds. C. Ciampi, F. Socci Natali and G. Taddei Elmi), pp. 681-693. CEDAM, Padova. Also published as report SKBS/B3.A/93-18 (Verheij, 1993).

Verheij, B. (1995e). *Rules and reasons: from ontology to rational standards for defeasible reasoning*. Published as report SKBS/B3.A/95-07.

Verheij, B. (1996a). Two approaches to dialectical argumentation: admissible sets and argumentation stages. Presented at the *Computational Dialectics Workshop* at FAPR-96. June 3-7, 1996, Bonn. Published as report SKBS/B3.A/96-01. An abstract is available on the World-Wide Web at http://www.cs.unimaas.nl/~verheij/papers/cd96.htm. The full text is available on the World-Wide Web at http://nathan.gmd.de/projects/zeno/fapr/programme.html.

Verheij, B. (1996b). An integrated view on rules and principles. To be presented at the *Ninth International Conference on Legal Knowledge-Based Systems (JURIX '96)*. Friday, December 13, 1996, Tilburg. Published as report SKBS/B3.A/96-05.

Verheij, B., and Hage, J.C. (1994). Reasoning by analogy: a formal reconstruction. *Legal knowledge based systems. The relation with legal theory* (eds. H. Prakken, A.J. Muntjewerff and A. Soeteman), pp. 65-78. Koninklijke Vermande, Lelystad. Also published as report SKBS/B3.A/94-14.

Visser, P.R.S. (1995). *Knowledge Specification for Multiple Legal Tasks. A Case Study of the Interaction Problem in the Legal Domain*. Doctoral thesis, Leiden University, Leiden.

Vreeswijk, G. (1991). Abstract argumentation systems: preliminary report. *Proceedings of the First World Conference on the Fundamentals of Artificial Intelligence* (eds. D.M. Gabbay and M. De Glas), pp. 501-510. Angkor, Paris.

Vreeswijk, G. (1993). *Studies in defeasible argumentation*. Doctoral thesis, Vrije Universiteit, Amsterdam.

Vreeswijk, G. (1995). Representation of Formal Dispute with a Standing Order. *MATRIKS Reports in Knowledge Engineering*. MAastricht Techological Research Institute for Knowledge and Systems, Maastricht. An abstract is available on the World-Wide Web at http://www.cs.unimaas.nl/~vreeswyk/abstracts/pps.htm.

Yoshino, H., Haraguchi, M., Sakurai, S., and Kagayama, S. (1993). Towards a legal analogical reasoning system: knowledge representation and reasoning methods. *The Fourth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 110-116. ACM, New York (New York).

Index

Summary

The subject of this thesis is argumentation. We consider argumentation as a process in which arguments supporting a conclusion are taken into account. During the process of argumentation, a conclusion originally justified by some argument can become no longer justified. This is the result of the *defeasibility* of arguments, a term introduced by Hart in 1948 (cf. Loui, 1995a). Our central theme is how argumentation and the defeasibility of arguments can be formally modeled.

The purpose of our research is to find answers to two groups of research questions.

- What is the role of rules and reasons in argumentation with defeasible arguments? What properties of rules and reasons are relevant for argumentation and defeat? How do these properties relate?
- What is the role of process in argumentation with defeasible arguments? How is the defeat of an argument determined by its structure, counterarguments and the argumentation stage?

Trying to answer these groups of questions, we study argumentation and defeat from two angles, resulting in formalisms of different nature, Reason-Based Logic and CumulA.

Reason-Based Logic is a model of the nature of rules and reasons, which are at the basis of argumentation. We investigate the properties of rules and reasons that are relevant for the argumentation and defeat, and how these properties relate to each other.

CumulA is a model of argumentation in stages. We investigate how the structure of an argument is related to defeat, when arguments are defeated by counterarguments, and how the status of arguments is affected by the argumentation stage.

The thesis has five goals:

- Providing a model of rules and reasons, Reason-Based Logic, focusing on properties that are relevant for the defeasibility of arguments.
- Demonstrating the usefulness of the model by providing examples in the field of law.
- Discussing how Reason-Based Logic relates to previously proposed models.
- Providing a model of argumentation, CumulA, that focuses on the process of constructing arguments, and shows how the status of an argument is

determined by the structure of the argument, the counterarguments and the stage of the argumentation process.
- Demonstrating how CumulA can be used to analyze other models of argumentation.

Each of these goals corresponds to a chapter. In chapter 2, we describe Reason-Based Logic. We determine types of facts concerning rules and reasons that are relevant for the defeasibility of arguments, and show their relations. Using this semantics of rules and reasons, we determine some intuitively attractive modes of reasoning. However, these lead to the difficulties of nonmonotonic reasoning. We show how the ideas of Reiter (1980, 1987) can be used to define rigorously which conclusions nonmonotonically follow from a given set of premises.

Chapter 3 contains a series of examples of Reason-Based Logic, taken from the field of law. We give applications of Reason-Based Logic to the theory of legal reasoning: we describe three different ways of reconstructing reasoning by analogy, and provide an integrated view on rules and principles, which seem fundamentally different (cf. Dworkin, 1978, p. 22ff. and 71ff.).

In chapter 4, we survey other models of rules, and compare them to Reason-Based Logic. We do this by treating a number of issues concerning the formalization of rules, and discussing various approaches to deal with these issues.

In chapter 5, the second part of the thesis starts with a discussion of CumulA. It is a formal model of argumentation with defeasible arguments, focusing on the process of taking arguments into account. The main ingredients of the formalism are arguments, defeaters, argumentation stages and lines of argumentation.

In chapter 6, we show how CumulA can be used to analyze models of argumentation. We investigate types of argument structure and of defeat, the role of inconsistency and counterarguments for defeat, and directions of argumentation. As a result, we are able to distinguish a number of existing argumentation models on formal grounds.

The thesis ends with the results and conclusions of the research (chapter 7). We also give some suggestions for future research.

The contributions of the thesis are as follows:

1. We have separated the semantics of rules and reasons, as used in argumentation with defeasible arguments, from the definition of a defeasible consequence relation. In this way, the definition of defeasible reasoning becomes less ad hoc, and is based on explicit standards (cf. chapter 2, section 6).
2. We have shown that it is advantageous to consider rules as special objects and to use a translation from sentences to terms (cf. chapter 2, section 4). In this way, it becomes possible to represent facts about rules, and to reason with them. As a result, we could keep the merits of two competing approaches: the use of rule identifiers and the use of special-purpose conditionals. Our approach enhances the ad hoc use of rule identifiers, that was introduced in order to represent facts about rules. At the same time, our approach can

represent the validity of rules, which is an advantage of the use of special-purpose conditionals in contrast with the use of rule identifiers (cf. chapter 4).

3. We have separated the generation of a reason and the generation of a conclusion, that both can occur when the condition of a rule is satisfied. First, this clarifies the relation of rules and reasons, and second this allows different levels where defeasibility can occur (cf. chapter 3, sections 5 and 6).
4. We have presented an integrated view on rules and principles, and have shown that rules and principles can be regarded as the extremes of a spectrum of hybrid rules/principles (cf. chapter 3, sections 2 and 7). This integrated view is in contrast with Dworkin's strict distinction between rules and principles (cf. Dworkin, 1978). The view is formally elaborated in Reason-Based Logic.
5. We have given three different ways of reconstructing reasoning by analogy (cf. chapter 3, section 8): (1) application of principles that underlie the original rule, (2) application of an analogous rule/principle that has the same underlying principles as the original rule, and (3) analogous application of the original rule, i.e., the application of the rule with non-standard justification.
6. We have shown how the effect of the accrual of reasons on the defeat of arguments can be dealt with in a formal model. In Reason-Based Logic, we focused on the weighing of sets of reasons (chapter 2); in CumulA, we focused on the coordination of arguments and defeat by parallel strengthening (chapter 5).
7. We have provided the model of argumentation CumulA, in which the defeat of arguments is determined by the structure of arguments, counterarguments, and the stage of the argumentation process. We have shown that CumulA's defeaters can represent a wide range of types of defeat, that were not previously integrated in one formalism (cf. chapter 5).
8. We have used CumulA to analyze argumentation models. First, we have made several formal distinctions between argumentation theories. Second, we have captured elements of a number of existing argumentation models in CumulA's argumentation theories. Third, we have applied the distinctions to the resulting argumentation theories. As a result, we were able to show similarities and differences between the argumentation theories capturing argumentation models by applying the formal distinctions above (chapter 6).

Samenvatting

Het onderwerp van dit proefschrift is argumentatie. We beschouwen argumentatie als een proces. Tijdens dit proces worden redeneringen geconstrueerd ter ondersteuning van een conclusie. Gedurende dit proces kan een conclusie aanvankelijk wel en later niet meer gerechtvaardigd zijn door een redenering. Dit komt door de weerlegbaarheid van redeneringen (Eng. defeasibility of arguments). Ons centrale thema is hoe argumentatie en de weerlegbaarheid van redeneringen formeel kan worden gemodelleerd.

Ons onderzoeksdoel is het vinden van antwoorden op de twee groepen onderzoeksvragen.

- Wat is de rol van regels en redenen in argumentatie met weerlegbare redeneringen? Welke eigenschappen van regels en redenen zijn relevant voor argumentatie en weerlegging? Hoe verhouden deze eigenschappen zich tot elkaar?
- Wat is de rol van het argumentatieproces bij argumentatie met weerlegbare redeneringen? Hoe wordt de weerlegging van een redenering bepaald door de structuur van de redenering, andere redeneringen, en het argumentatiestadium?

Ter beantwoording van de vragen bestuderen we argumentatie en weerlegging vanuit twee gezichtspunten. Dit leidt tot formalismen van verschillende aard, Reason-Based Logic en CumulA.

Reason-Based Logic is een model van de aard van regels en redenen, die de basis vormen van argumentatie. We onderzoeken welke eigenschappen van regels en redenen relevant zijn voor argumentatie en weerlegging, en hoe deze eigenschappen zich tot elkaar verhouden.

CumulA is een model van argumentatie in stadia. We onderzoeken hoe de structuur van een redenering zich verhoudt tot weerlegging, wanneer andere redeneringen een redenering weerleggen, en hoe het argumentatiestadium de status van een redenering beïnvloedt.

Het proefschrift heeft vijf doelen:

- Het beschrijven van een model van regels en redenen, Reason-Based Logic, gericht op eigenschappen die relevant zijn voor de weerlegging van redeneringen.
- Het aantonen van de bruikbaarheid van het model door het geven van juridische voorbeelden.

- Het laten zien van de verbanden van Reason-Based Logic met eerdere voorgestelde modellen.
- Het beschrijven van een argumentatiemodel, CumulA, dat gericht is op het proces van het construeren van rederingen en dat laat zien hoe de status van een redenering wordt bepaald door de structuur van de redenering, andere redeneringen, en het argumentatiestadium.
- Het aantonen van de bruikbaarheid van CumulA bij het analyseren van andere argumentatiemodellen.

Elk doel wordt behandeld in een hoofdstuk. In hoofdstuk 2 beschrijven we Reason-Based Logic. We bepalen feittypen voor regels en redenen die relevant zijn voor de weerlegbaarheid van argumentatie, en laten de relaties tussen de feittypen zien. Gebruik makend van deze semantiek van regels en redenen bepalen we enkele intuïtief aantrekkelijke redeneerwijzen. Deze redeneerwijzen leiden echter tot de problemen van niet-monotoon redeneren. We laten zien hoe de ideeën van Reiter (1980, 1987) kunnen worden gebruikt voor een formele definitie van de conclusies die de niet-monotone gevolgen zijn van gegeven premissen.

Hoofdstuk 3 bevat een reeks voorbeelden van Reason-Based Logic in het recht. We geven twee toepassingen van Reason-Based logic in de rechtstheorie. Ten eerste bechrijven we drie manieren om redeneren naar analogie te reconstrueren. Ten tweede geven we een geïntegreerde kijk op regels en beginselen, die fundamenteel van elkaar lijken te verschillen (cf. Dworkin, 1978, p. 22ff. en 71ff.).

In hoofdstuk 4 geven we een overzicht van modellen van regels en vergelijken ze met Reason-Based Logic. We doen aan de hand van een aantal problemen bij de formalisering van regels te en behandelen benaderingen om met deze problemen om te gaan.

In hoofdstuk 5 begint het tweede deel van het proefschrift met de beschrijving van CumulA. Het is een formeel model van argumentatie met weerlegbare redeneringen, gericht op het geleidelijk construeren van redeneringen. De belangrijkste ingrediënten van het formalisme zijn redeneringen, weerleggers (Eng. defeaters), argumentatiestadia en betogen (Eng. lines of argumentation).

In hoofdstuk 6 laten we zien hoe CumulA gebruikt kan worden voor het analyseren van argumentatiemodellen. We onderzoeken typen redeneringen en weerlegging aan de hand van de structuur van redeneringen, de rol van inconsistentie en tegenargumenten in weerlegging en argumentatierichtingen. Zo kunnen we een aantal bestaande argumentatiemodellen op formele gronden van elkaar onderscheiden.

Het proefschrift eindigt met de resultaten en conclusies van het onderzoek (hoofdstuk 7). We geven ook enkele suggesties voor toekomstig onderzoek.

De bijdragen van het proefschrift zijn als volgt:

1. We hebben de semantiek van regels en redenen, zoals die gebruikt worden in argumentatie met weerlegbare redeneringen, onderscheiden van de definitie van een weerlegbare-gevolgtrekkingsrelatie. Zo wordt de definitie van

weerlegbaar redeneren minder ad hoc en kan ze gebaseerd worden op expliciete standaarden (cf. chapter 2, section 6).

2. We hebben laten zien dat het voordelig is om regels als speciale objecten te beschouwen en een vertaling tussen zinnen en termen te gebruiken (cf. chapter 2, section 4). Zo wordt het mogelijk om feiten over regels te representeren en over regels te redeneren. Als gevolg hiervan konden de voordelen van twee benaderingen worden behouden: het gebruik van regelnamen en het gebruik van een speciale conditionele zinsstructuur. Onze benadering verbetert het gebruik van regelnamen, dat was geïntroduceerd om feiten over regels te representeren. Tegelijkertijd kan onze benadering de geldigheid van regels representeren. Dit was een voordeel van het gebruik van een speciale conditionele zinsstructuur tegenover het gebruik van regelnamen. (cf. chapter 4).
3. We hebben het onstaan van een reden en het trekken van een conclusie van elkaar gescheiden. Beide kunnen plaatsvinden als aan de voorwaarde van een regel is voldaan. Ten eerste verheldert dit de relatie tussen regels en redenen en ten tweede kan weerlegging op verschillende niveaus voorkomen (cf. chapter 3, sections 5 and 6).
4. We hebben een geïntegreerde kijk op regels en beginselen gegeven en laten zien dat regels en beginselen beschouwd kunnen worden als de extremen van een spectrum van hybride regels/beginselen (cf. chapter 3, sections 2 and 7). Deze geïntegreerde kijk contrasteert met Dworkin's stricte onderscheid tussen regels en beginselen (cf. Dworkin, 1978). De kijk wordt formeel uitgewerkt in Reason-Based Logic.
5. We hebben drie manieren beschreven om redeneren naar analogie te reconstrueren (cf. chapter 3, section 8): (1) als de toepassing van beginselen die aan de oorspronkelijke regel ten grondslag liggen, (2) als de toepassing van een analoge regel of beginsel met dezelfde onderliggende beginselen als de oorspronkelijke regel, en (3) de analoge toepassing van de oorspronkelijke regel, d.w.z. de toepassing van de regel met niet-standaard rechtvaardiging.
6. We hebben laten zien hoe met het effect van de ophoping van redenen (Eng. accrual of reasons) op de weerlegging van redeneringen formeel kan worden omgegaan. In Reason-Based Logic waren we gericht op het wegen van verzamelingen redenen (chapter 2); in CumulA waren we gericht op de nevenschikking van redeneringen en weerlegging door parallele versterking (chapter 5).
7. We hebben het argumentatiemodel CumulA voorgesteld. Hierin wordt de weerlegging van redeneringen bepaald door hun structuur, door andere argumenten en door het argumentatiestadium. We hebben laten zien dat CumulA's weerleggers (Eng. defeaters) een breed scala van typen weerlegging kunnen representeren. Deze typen zijn nog niet eerder in een formalisme geïntegreerd (cf. chapter 5).
8. We hebben CumulA gebruikt om bestaande argumentatiemodellen te analyseren. Eerst hebben we een aantal formele onderscheidingen gemaakt

voor CumulA's argumentatietheorieën. Daarna hebben we elementen van een aantal bestaande argumentatiemodellen beschreven in CumulA's argumentatietheorieën. Tenslotte hebben we de gemaakte onderscheidingen toegepast op deze argumentatietheorieën. Op deze manier was het mogelijk om op grond van de genoemde formele onderscheidingen overeenkomsten en verschillen tussen deze argumentatietheorieën te laten zien.

Curriculum Vitae

Bart Verheij was born in Hoorn, The Netherlands, in 1967. From 1979 to 1985, he attended the Johan van Oldenbarnevelt Gymnasium in Amersfoort. He studied mathematics at the University of Amsterdam, specializing in algebraic geometry. In his final year, he was a student assistant at the Department of Mathematics. In the summer of 1991, he wrote a report on the cohomology of moduli spaces of curves, related to Witten's research in theoretical physics. In 1992, he started working as a Ph.D. researcher (in Dutch: assistent in opleiding) at the Universiteit Maastricht, then called the Rijksuniversiteit Limburg, in Maastricht, The Netherlands. He worked at the Departments of Computer Science and of Metajuridica. His research was part of the ARCHIMEDES project (SKBS/B3.A). The project was financially supported by the Foundation for Knowledge-Based Systems (SKBS). He started doing multi-media information retrieval research. Gradually his attention shifted to theoretical research on argumentation and defeat. In 1995, he co-organized the Eighth International Conference on Legal Knowledge-Based Systems (JURIX '95). He was invited to work at the Computer Science Department of the Washington University in Saint Louis (Missouri) in the summer of 1996. His stay there was sponsored by the National Science Foundation (NSF) under grant number 9503476. In 1997, he starts working in a project on the application of the recent theoretical insights into legal reasoning to the legal domain of tort. This project is sponsored by the Dutch National Programme for Information Technology and Law (ITeR).