Argument-based extended logic programming with defeasible priorities

Henry Prakken*
Computer/Law Institute
Faculty of Law, Free University Amsterdam
De Boelelaan 1105, 1081 HV Amsterdam
The Netherlands
email: henry@rechten.vu.nl

Giovanni Sartor
Faculty of Law
The Queen's University of Belfast
Belfast BT7 1NN, Northern Ireland
email: sartor@cirfid.unibo.it

ABSTRACT. Inspired by legal reasoning, this paper presents a semantics and proof theory of a system for defeasible argumentation. Arguments are expressed in a logic-programming language with both weak and strong negation. Conflicts between arguments are decided with the help of priorities on the rules. An important feature of the system is that these priorities are not fixed, but are themselves defeasibly derived as conclusions within the system. Thus debates on the choice between conflicting arguments can also be modelled. The semantics of the system is given with a fixpoint definition, while its proof theory is stated in dialectical style, where a proof takes the form of a dialogue between a proponent and an opponent of an argument: an argument is shown to be justified if the proponent can make the opponent run out of moves in whatever way the opponent attacks.
KEYWORDS: Argumentation, Nonmonotonic reasoning, Extended logic programming, Legal reasoning, Defeasible priorities

Introduction

When knowledge-based systems have to cope with uncertain, incomplete or inconsistent information, various strategies might be appropriate, depending on the context of application. When during the lifetime of a system better

*Supported by a research fellowship of the Royal Netherlands Academy of Arts and Sciences, and by Esprit WG 8319 'Modelage'.

information may become available, it might be useful to allow for revisions of the knowledge base. And when the system has to act in a changing world, where information can be outdated by new events, it might be good to allow for updates of the system. However, sometimes the uncertainty, incompleteness or inconsistency of information is not caused by the costs or impossibility of finding better information, or by changes in the world, but by the inevitable existence of disagreement and conflict of opinion in the domain of application. An example of such a domain is the law, where different agents are involved, with different interests, opinions and responsibilities. Accordingly, a system that, say, assists a lawyer in preparing a lawsuit has to reflect the conflicts of interest and political, moral or social opinions that in legal contexts inevitably arise. The same holds for a system that supports decision making and negotiation in bureaucratic organisations. If a system is applied in such contexts, trying to improve the knowledge base does not make much sense. Rather the system should be able generate useful information from an imperfect knowledge base: it should be able to record or produce alternative arguments for and against a certain claim, and it should be able to reason with information on the strengths and weaknesses, or on the hierarchical status, of the various opinions that are put forward. The aim of this paper is to contribute to the logical study of these phenomena, by presenting an argument-based system for defeasible reasoning, with a logic-programming-like language, and with defeasible priorities.

Argument-based systems analyze defeasible, or nonmonotonic reasoning in terms of the interactions between arguments for alternative conclusions. Nonmonotonicity arises from the fact that arguments can be defeated by stronger counterarguments. This approach has proved to be a fruitful paradigm for formalising defeasible reasoning; cf. [POL 87, Loui 87, Nute 88, SL 92, VRE 93a, PRA 93, BTK 93, Dung 95]. Not only does the notion of an argument naturally point at possible proof theories, but also do notions like argument, counterargument, attack, rebuttal and defeat have natural counterparts in the way people think, which makes argument-based systems transparent in applications. Especially in legal reasoning these notions are prevalent, which explains why several argument-based systems have been applied to that domain ([PRA 91, Loui 93, SAR 94, LN 95, GOR 95, PS 96b]).

Also the present system is inspired by the legal domain (and similar domains, such as bureaucracies). Firstly, we want to capture that in law the criteria for comparing arguments are themselves part of the domain theory and thus debatable (defeasible). For instance, most legal systems have a principle reflecting their hierarchical structure (e.g. 'the constitution has priority over statutes'), a temporal principle that the later rule overrides the earlier one, and a legal version of the specificity principle. But also many less abstract principles can be found. For instance, in Italian town planning regulations we can find a priority rule stating that rules on the protection of artistic buildings prevail over rules concerning town planning. Apart from varying from domain to domain, these priority rules can also be debatable, in the same way as 'ordinary' domain information can be. For instance, debates are possible about the relation between the hierarchical, temporal and specificity principle. Or if in Italy an artistic-buildings rule is of an earlier date than a conflicting town planning rule, the town-planning conflict rule is in conflict with the temporal principle. Other conflict rules may apply to such conflicts, and this makes that also reasoning about priorities is defeasible.

Secondly, we want to respect that in law specificity is not the overriding standard for comparing arguments. In most legal systems it is subordinate to the hierarchical and the temporal criterion, and sometimes also to other criteria. This means that systems like [GP 92], making specificity the overriding standard for comparison, are for our purposes inadequate.

Finally, we want to model the fact that legal reasoning combines the use of priorities to choose between conflicting rules with the use of assumptions, or 'weak' negation, to express explicit exceptions. For instance, section 3:32–(1) of the Dutch Civil Code declares every person to have the capacity to perform juridical acts, "unless the law provides otherwise". And section 2 of the Dutch rent act declares the rest of that act inapplicable to leases that by their nature are short-termed. Accordingly, our language will have both strong and weak negation, which yields two different ways of attacking an argument: by stating an argument with a contradictory conclusion, or by stating an argument with a conclusion providing an 'unless' clause of the other argument. The reason why we want this complexity in our system, and do not reduce priorities to assumptions or vice versa, as in many other nonmonotonic logics, is that we want to allow for formalisations that preserve the surface structure of legal texts as much as possible; this in turn might facilitate a modular formalisation process.

It is not our aim to present a general theory of defeasible argumentation. Rather, we will analyze these phenomena within a logic-programming-like setting, in particular Dung's [Dung 93b] argument-based approach to the semantics of extended logic programming. With the choice for a logic-programming like language we hope to increase the potential for implementation, while our choice for Dung's approach is motivated by his emphasis on argumentation. We will develop both a semantics and a proof theory for our system. The semantics, which is of a sceptical nature, is given with a fixpoint definition, while the proof theory is stated in dialectical style, where a proof takes the form of a dialogue between a proponent and an opponent of an argument: an argument is shown to be justified if the proponent can make the opponent run out of moves in whatever way the opponent attacks.

We will start in Section 1 with an informal sketch of argument-based systems. Then we will present our own system, in two phases. In the first phase the priorities are still externally given and fixed (Sections 2, 3 and 4) and in the second phase they are derived within the system itself (Sections 5 and 6). In Section 7 we briefly investigate an alternative, credulous semantics of our system. We end with some remarks on implementation (Section 8), a comparison with related work (Section 9) and some concluding remarks. Proofs can be found in the appendix.

This paper is a revised and extended version of [PS 96a]. A less technical companion paper is [PS 96b], which presents and applies the system (in its dialectical form) in a legal context.

1 Argumentation Frameworks: General Ideas

Our system takes the form of a system for defeasible argumentation, or an 'Argumentation Framework' (AF). In this section we discuss the general ideas behind AF's. Most such systems have been developed in general Artificial-Intelligence research on nonmonotonic reasoning, although Pollock's work on defeasible argumentation, e.g. in [POL 87], was developed to analyze epistemological issues in the philosophy of science. In AI argument-based systems have been developed as a reformulation of, [BTK 93, Dung 95], or an alternative to, [Loui 87, SL 92, VRE 93a], earlier formalisations of nonmonotonic reasoning. The idea here is that nonmonotonic reasoning can be analyzed in terms of the interactions between arguments for alternative conclusions. Nonmonotonicity, or defeasibility, arises from the fact that arguments can be defeated by stronger counterarguments.

To describe the general structure of AF's, they contain the following five elements, although sometimes implicitly. AF's have an underlying logical *language*, notions of an *argument* and of *conflicts* between arguments, a relation of *defeat* among arguments and, finally, a definition of the *status* of arguments. To discuss these elements, AF's are built around an underlying logical system for the construction of individual arguments. Some AF's, like ours, assume a particular logic, while other systems, e.g. [VRE 93a, BTK 93, Dung 95], leave the underlying logic partly or wholly unspecified; thus these systems can be instantiated with various alternative logics, as [Dung 93b] has done for extended logic programming. Then AF's have the notion of an argument, which corresponds to a proof in the underlying logic. This is a narrow use of the term 'argument', which should not be confused with the broader meaning it often has in AI and philosophy, when it also includes, for instance, inductive, abductive and analogical arguments.

The notions of an underlying logic and an argument still fit with the standard picture of what a logical system is. The remaining three elements are what makes an AF a framework for adversarial argumentation. The first is the notion of a conflict between arguments. The typical case is when arguments have contradictory conclusions. An AF also has ways of comparing arguments, on the basis of certain standards, to see whether an argument defeats a counterargument. In AI the specificity principle is regarded as very important but, as said above, one of the main themes of this paper is that any standard might be used: their content is part of the domain theory, and is debatable, just as the rest of the domain theory is.

Since attacking arguments can themselves be attacked by other arguments, comparing pairs of arguments is not sufficient; what is also needed is a definition of the status of arguments on the basis of all ways in which they interact.

It is this definition that produces the output of an AF: it divides arguments in, typically, three classes: arguments with which a dispute can be 'won', respectively, 'lost' and arguments which leave the dispute undecided. In this paper we will denote these classes with the terms 'justified', 'overruled' and 'defensible' arguments.

2 The Formal System I: Fixed Priorities

We now present our own argumentation framework, along the lines of the just-given informal description. We will present the system in two stages: in this and the following two sections we will assume that the priorities are fixed and undisputable, while in section 5 we will make them defeasible, by allowing also arguments about the priorities.

2.1 The Language

The object language of our system is of familiar logic-programming style: it contains a twoplace one-direction connective that forms rules out of literals, which can contain two kinds of negation, weak and strong negation (weak negation is also often called 'negation as failure', while strong negation is sometimes called 'classical' or 'explicit' negation). A *strong* literal is an atomic first-order formula, or such a formula preceded by strong negation $\neg$. For any atom A we say that A and $\neg A$ are the complement of each other; in the metalanguage we will denote the complement of a literal L with $\overline{L}$. A *weak* literal is a literal of the form $\sim L$, where L is a strong literal. Informally, $\sim L$ reads as 'there is no evidence that L is the case', while $\neg L$ says 'L is definitely not the case'.[1]

Definition 2.1 *A* rule *is an expression of the form*

$$r : L_0 \wedge \ldots \wedge L_j \wedge \sim L_k \wedge \ldots \wedge \sim L_m \Rightarrow L_n$$

where r, a first-order term, is the name of the rule and each L_i $(0 \leq i \leq n)$ is a strong literal. The conjunction at the left of the arrow is the antecedent *and the literal at the right of the arrow is the* consequent *of the rule. As usual, a rule with variables is a scheme standing for all its ground instances.*

Note that a rule may have zero literals in its antecedents. Such a rule can be used to express an unconditional statement.

Variables will in this paper be denoted with the letters x, y and z, and constants with any other letter, or with words. To enhance readability of examples, we will often write literals in a casual style. For instance, 'x is earlier than y' is our variant of the more standard logical notation $Earlier(x, y)$.

[1] Note that compared to some other articles in this issue we switch the symbols $\neg$ and $\sim$. This is to keep in line with related work of ourselves and others on extended logic programming.

Next we define the input information of our system, which will be called an *ordered theory*. Obviously, the input contains rules. However, we will assume that the input rules are divided into two categories, the set D of *defeasible* rules and the set S of *strict* rules; we assume that no member of S contains a weak literal. Informally, the defeasible rules express information that is intended by the user of the system to be subject to debate, while the strict rules represent the information that is intended to be beyond debate. Note that a defeasible rule can be defeasible in two ways: it can be overridden by stronger rules with a contradicting consequent, and it can contain assumptions that are not warranted. The strict rules can be used to express meaning postulates, like 'Bachelors are not married', and properties of relations, like transitivity of the 'older than' relation or asymmetry of the 'father of' relation. Technically the role of the strict rules is roughly the same as, for instance, the first-order part of a default theory in default logic.

Syntactically, strict and defeasible rules differ in only one respect: only defeasible rules can contain assumptions. For the rest we have chosen to make the distinction between strict and defeasible rules not at the syntactic but at the pragmatic level. That is, the logic governing the conditional operator is the same for both types of rules; they differ only in their epistemological status. Yet, for notational convenience we will, when writing down rules, use $\rightarrow$ when a rule is strict and $\Rightarrow$ when a rule is defeasible. However, formally the symbol $\rightarrow$ is not part of our object language; it is just a shorthand for saying that a rule is in the set of strict rules.

As discussed in the introduction, in legal reasoning conflicts between defeasible rules are often resolved with the help of statements on which rule takes precedence. Accordingly, our input information will also contain an ordering $<$ on the set D, which we will assume to be a strict partial order, i.e. transitive (if $x < y$ and $y < z$ then $x < z$) and asymmetric (if $x < y$ then $y \not< x$). '$x < y$' means that y is preferred over x. To capture that strict rules are beyond debate, no ordering needs to be defined on S.

In sum, our system assumes input information in the form of an *ordered theory* $(S, D, <)$, where S and D are sets of, respectively, strict and defeasible rules and $<$ is a strict partial order on D.

2.2 Arguments

The basic notion of a system for defeasible argumentation is that of an argument. In general, the idea is that an argument for a certain proposition is a proof of that proposition in the logic of the underlying language. In our system, the simple language gives rise to a simple notion of an argument, viz. as a sequence of rules that can be chained together, and that is 'grounded' in the facts. This is captured by a slight variant of [Dung 93b]'s notion of a 'defeasible proof'.

Definition 2.2 *An* argument *is a finite sequence $A = [r_0, \ldots, r_n]$ of ground instances of rules such that*

1. *for every i $(0 \leq i \leq n)$, for every strong literal L_j in the antecedent of r_i there is a $k < i$ such that L_j is the consequent of r_k;*

2. *No two distinct rules in the sequence have the same consequent.*

An argument A is based on *the ordered theory $(S, D, <)$ iff all rules of A are in $S \cup D$.*[2]

Condition (1) says that arguments are formed by chaining rules together. A subtlety is that in doing so, weak literals may be ignored. In fact, this condition assumes two implicit inference rules: a rule for conjoining strong literals and a rule of 'defeasible modus ponens', of the form

$$\frac{\begin{array}{ll} r: & L_0 \wedge \ldots \wedge L_j \wedge \sim L_k \wedge \ldots \wedge \sim L_m \Rightarrow L_n, \\ & L_0 \wedge \ldots \wedge L_j \end{array}}{L_n}$$

Note that Condition (2) prevents arguments from containing circular chains of rules.

For any ordered theory Γ we will denote the set of all arguments on the basis of Γ with $Args_\Gamma$. Likewise, for any set R of rules, $Args_R$ stands for the set of all arguments that consist of only rules in R.

We will also use the following notions.

Definition 2.3 *For any argument A:*

1. *A is* strict *iff it does not contain any defeasible rule; it is* defeasible *otherwise.*

2. *An argument A' is a* (proper) subargument *of A iff A' is a (proper) subsequence of A.*

3. *A literal L is a* conclusion *of A iff L is the consequent of some rule in A.*

4. *A literal L is an* assumption *of A iff $\sim \overline{L}$ occurs in some rule in A.*

Condition (3) says that the consequent of any rule in the chain is a conclusion of the argument. The alternative, to regard only the consequent of the last rule as the conclusion, is perhaps philosophically more usual, but our choice makes some other definitions simpler.
To illustrate these notions, for the (defeasible) argument

$$A = [r_1\colon \rightarrow a,\ r_2\colon\ a \wedge \sim \neg b \Rightarrow c,\ r_3\colon\ c \wedge \sim d \Rightarrow e]$$

A's conclusions are $\{a, c, e\}$, its subarguments are $\{[\,], [r_1], [r_1, r_2], [r_1, r_2, r_3]\}$, of which only $[\,]$ and $[r_1]$ are strict, and A's assumptions are $\{b, \neg d\}$.

[2]Where there is no danger of confusion, we will leave the phrase 'based on $(S, D, <)$' implicit.

Example 2.4 The following example illustrates the distinction between weak and strong negation. Rule r_1 states that a person who cannot be shown to be a minor has the capacity to perform legal acts, and rule r_2 requires that a person is positively shown not to be a minor, in order that s/he can exercise the right to vote. The point is that these rules give rise to an argument for 'x has legal capacity' but not for 'x has the right to vote', since there is no rule providing the antecedent of r_2.

r_1: $\sim x$ is a minor $\Rightarrow x$ has legal capacity
r_2: $\neg\ x$ is a minor $\Rightarrow x$ has the right to vote

2.3 Conflicts between Arguments

So far the notions have been fairly standard; now we will present the adversarial aspects of our system. We first define in which ways an argument can attack, i.e. be a counterargument of another argument. This definition does not yet include any way of checking which argument is better; it only tells us which arguments are in conflict. The two different kinds of negation give rise to two ways of attacking an argument. Let us first explain them informally. Assume an argument A has a conclusion L. Then the first way of attack consists in contradicting a conclusion: i.e. A attacks any argument B which has a conclusion $\overline{L}$. Clearly, such a head-to-head conflict is symmetric: if A head-to-head attacks B, then B head-to-head attacks A. The second way of attack, with weak negation, consists in contradicting an assumption: i.e. A attacks any argument B which has an assumption $\overline{L}$. Obviously, this way of attack is not symmetric. It can be seen as a special case of [POL 87]'s 'undercutting defeaters'. See further Section 9.

However, strict rules complicate the matter. To see this, consider the following example.

Example 2.5 Assume we have two defeasible rules

$r_1 : A \Rightarrow x$ is a lease
$r_2 : B \Rightarrow \neg\ x$ is a contract

and a strict rule

$s_1 : x$ is a lease $\rightarrow x$ is a contract

and assume we have the strict facts f_1: $\rightarrow A$, f_2: $\rightarrow B$. Now the rules with complementary consequents are[3] r_2 and s_1. However, intuitively it seems that r_2 does not compete with the strict rule s_1 but with the defeasible rule r_1; s_1 just expresses a linguistic convention.

Here is a variant of this example.

[3]Sometimes we will leave implicit that we refer to ground instances of rules.

Example 2.6 Assume we have

$r_1 : A \Rightarrow x$ is married
$r_2 : B \Rightarrow x$ is a bachelor
$s_1 : x$ is married $\rightarrow \neg\ x$ is a bachelor
$s_2 : x$ is a bachelor $\rightarrow \neg\ x$ is married

and the strict facts f_1: $\rightarrow A$, f_2: $\rightarrow B$. Again, the rules that intuitively are in conflict with each other are not the rules with complementary consequents (r_1 vs. s_2 and r_2 vs. s_1), but the two defeasible rules r_1 and r_2. The strict rules just declare the predicates 'married' and 'bachelor' to be incompatible.

These examples are the reason why we deviate from [Nute 88, Nute 94], which system, like ours, has two categories of one-direction rules, although no weak negation. Nute only regards rules with complementary consequents as conflicting and therefore he regards in the examples 2.5 and 2.6 the arguments $[s_1, r_1]$ and $[s_2, r_2]$ as not conflicting. This makes that the priority relation between r_1 and r_2 is not taken into account when deciding the conflicts between two arguments that someone is, and is not a bachelor (or married).

We propose the following way of defining a conflict between arguments: if an argument A has a conclusion or assumption L, we should consider A as being attacked by all those arguments which can *with only strict rules* be extended to an argument with conclusion $\overline{L}$.

To illustrate this, in Example 2.5 the argument $[f_1, r_1]$ attacks the argument $[f_2, r_2]$, since $[f_1, r_1]$ can with the strict rule s_2 be extended to an argument for 'x is a contract'. More generally, in the first way of attack we have to extend *both* arguments with only strict rules, to see if eventually they will have complementary conclusions. This is captured by the following definitions, which, like all other definitions, are implicitly assumed to be relative to an ordered theory $(S, D, <)$. First we have to define the notion of extending an argument.

Definition 2.7 *Let A be an argument and T a sequence of rules. Then $A+T$ is the concatenation of A and T.*[4]

Now the notion of attacking/conflicting/counterarguments is defined as follows.[5]

Definition 2.8 *Let A_1 and A_2 be two arguments. Then A_1* attacks *A_2 iff there are sequences S_1, S_2 of strict rules such that $A_1 + S_1$ is an argument with conclusion L and*

1. $A_2 + S_2$ is an argument with a conclusion $\overline{L}$; or

[4]Since the order of rules does not matter except that the antecedents of a rule should come from a previous rule, we ignore the fact that $A + T$ is not uniquely defined.

[5]To prevent terminological confusion, it should be noted that [Dung 93b] and [BTK 93] use the term 'attack' in a different way, coming closer to our notion of 'defeat', to be defined below. They do not have an explicit notion of what we call 'attack'.

2. A_2 is an argument with an assumption $\overline{L}$.

We have already discussed some peculiarities of this definition when strict rules are involved. We can add that if two arguments contain defeasible rules with conflicting consequents, the definition is satisfied if S_1 and S_2 are empty. We now give some examples of this type, illustrating some more features of Definition 2.8.

Example 2.9 The first example shows that in order to attack an argument, a counterargument can point its attack at that argument itself, but also at one of its proper subarguments, thereby indirectly attacking the entire argument. So if we have

r_1: $\Rightarrow f$ forged evidence e
r_2: f forged evidence $e \Rightarrow \neg\ e$ is admissible evidence
r_3: $\Rightarrow \neg\ f$ forged evidence e

we have that $[r_3]$ does not only attack $[r_1]$, but also $[r_1, r_2]$.

Example 2.10 And if we vary this example into

r_1: $\Rightarrow f$ forged evidence e
r_2: f forged evidence $e \Rightarrow \neg\ f$ is honest
r_3: f is police officer $\Rightarrow f$ is honest
r_4: f is honest $\Rightarrow \neg\ f$ forged evidence

together with the fact

s_1: $\rightarrow f$ is police officer

then we have that $[r_1, r_2]$ does not only attack $[s_1, r_3]$ but also $[s_1, r_3, r_4]$. And $[s_1, r_3, r_4]$ does not only attack $[r_1]$, but also $[r_1, r_2]$.

The concept of 'attack/counterargument' is very important, since clearly a minimum requirement on any system for defeasible argumentation is that if two arguments are in conflict with each other, they cannot both be accepted as justified: at best, they can be accepted alternatively. We want our system to agree with this, in the sense that the set of 'justified arguments', to be defined later, is unique and conflict-free. To this end, we now define the following notions.

Definition 2.11 *An argument is* coherent *iff it does not attack itself.*

Two examples of incoherent arguments are $[r_1\colon\ \Rightarrow a,\ r_2\colon\ a \Rightarrow \neg a]$ and $[r_1\colon \sim a\ \Rightarrow b,\ r_2\colon\ b \Rightarrow a]$.

Definition 2.12 *A set Args of arguments is* conflict-free *iff no argument in Args attacks an argument in Args.*

2.4 Defeat among Arguments

Now that we know which arguments are in conflict with each other, the next step is to compare conflicting arguments. To this end we define, in two steps, a binary relation of defeat among arguments.[6] It is important to realise that this comparison does not yet determine with which arguments a dispute can be won; it only tells us something about the relation between two individual arguments (and their subarguments).

To capture the different 'force' of the two ways of attack, we define 'defeat' in terms of two other evaluative notions, rebutting and undercutting an argument. Their difference depends on whether the attacking argument contradicts a conclusion or an assumption of another argument. Only in the first case will the priorities be used. We will defend this choice after we have presented the definition of defeat, which will state how to use the notions of rebutting and undercutting an argument in combination.

Let us first concentrate on head-to-head conflicts and see how the priorities can be used to compare the conflicting arguments. In its simplest form, with a head-to-head conflict between two defeasible rules, we will compare the arguments with respect to the priority relation between these two rules (for a defence of this approach over comparing the arguments with respect to their 'weakest links', the reader is referred to [PRA 91, PRA 93]). However, if the conflict involves strict rules, then each argument may contain more than one rule of which the priority is relevant. To see this, consider $A = [r_1\colon \Rightarrow\ Rab, r_2\colon \Rightarrow\ Rbc]$, $B = [r_3\colon\ \Rightarrow Rca]$, and assume that S contains transitivity and asymmetry axioms for R. Then A can with these axioms be extended to an argument for $\neg Rca$, which means that A attacks B and vice versa. Now if we examine A, it seems natural to regard the priority of both r_1 and r_2 as relevant to deciding the conflict. So we want to compare r_3 with the set of rules $\{r_1, r_2\}$. In general, both arguments involved in a conflict can have more than one rule directly relevant to the conflict, and therefore in general we have to compare two *sets* of defeasible rules.

So what are the rules that are relevant for the comparison? In the simple case, when the conflict is between two defeasible rules with complementary consequents, they are exactly those defeasible rules. In the more complex case, when the conflict is determined by strict rules, they are, for each argument, all defeasible rules that are used to satisfy the antecedents of the strict rules with which the argument is (hypothetically) extended to establish a conflict. In the special case when the conflict is entirely caused by strict rules, the sets of relevant defeasible rules will be empty. Formally:

Definition 2.13 *For any argument A and set S of strict rules such that $A+S$ is an argument with conclusion L we define the set $R_L(A+S)$ of the defeasible rules relevant to L in the argument $A+S$ as follows.* $R_L(A+S) =$

[6] Note that the argument relations and properties defined below are relative to an implicitly assumed ordered theory.

1. *$\{r_d\}$ iff A includes a defeasible rule r_d with consequent L;*

2. *$R_{L_1}(A+S) \cup \ldots \cup R_{L_n}(A+S)$ iff A is defeasible and S includes a strict rule r_s: $L_1 \wedge \ldots \wedge L_n \rightarrow L$.*

Observe that since by definition no argument contains two rules with the same consequent, the set $R_L(A+S)$ is always unique.

To illustrate this definition, in the examples 2.6 and 2.10 the rules to be compared are r_1 and r_2. Consider next $A = [r_1\text{: } \Rightarrow Rab,\ r_2\text{: } \Rightarrow Rbc,\ s_1\text{: } Rab \wedge Rbc \rightarrow Rac]$. Then $R_{Rac}(A) = \{r_1, r_2\}$. However, if we make s_1 a defeasible rule, then $R_{Rac}(A) = \{s_1\}$.

Next we define how sets of rules are ordered on the basis of an ordering on their members.

Definition 2.14 *For any two sets R and R' of defeasible rules, $R < R'$ iff for some $r \in R$ and all $r' \in R'$ it holds that $r < r'$.*

The intuitive idea behind this definition is that if $R < R'$, R can be made better by replacing some rule in R with any rule in R', while the reverse is impossible.

Now we have enough tools to give the definition of rebutting and undercutting arguments. In fact, it will be almost the same as the one of attack; the only difference is that in the first case of attack, when arguments have complementary conclusions, we will use the priorities to assess them. Since we will not use the priorities in the second case, this means that an attack on an assumption always succeeds.

Definition 2.15 *Let A_1 and A_2 be two arguments. Then*

- A_1 rebuts A_2 *iff condition (1) of Definition 2.8 holds, provided that $R_L(A_1+S_1) \not< R_{\overline{L}}(A_2+S_2)$.*
- A_1 undercuts A_2 *iff condition (2) of Definition 2.8 holds.*

To inspect some properties of the definition, note first that from the definition of attack these definitions inherit that also rebutting and undercutting an argument can be 'direct', or 'indirect'. Let us now investigate some other properties. Observe first that if A rebuts or undercuts B, then A attacks B. Obviously, it does not hold that if A undercuts B, then B undercuts A. However, it also does not hold that if A undercuts B, then B does not undercut A. A counterexample is $A = [r_1\text{: } \sim b \Rightarrow a], B = [r_2\text{: } \sim a \Rightarrow b]$. It is also not the case that if A rebuts B, then B rebuts A. Just assume we have $A = [r_3\text{: } \Rightarrow a]$, $B = [r_4\text{: } \Rightarrow \neg a]$, and $r_3 < r_4$. However, rebutting involving a $<$ relation between conflicting rules is not always one-directional. To see this, assume in Example 2.10 that $r_1 < r_4$ and $r_3 < r_2$; then $[r_1, r_2]$ and $[s_1, r_3, r_4]$ rebut each other.

Finally we can give our definition of defeat. It states how in evaluating arguments the notions of undercutting and rebutting attack are combined. As a border case it regards any incoherent argument as defeated by the empty argument. Apart from this, the definition is based on two ideas, of which the first is implicit, and inherited from the definitions of attack, rebutting and undercutting: defeat of an argument can be direct, or indirect, by defeating one of its strict subarguments. The other idea is that attacks on assumptions do not only always succeed, but are also always stronger than an attack on a conclusion: if one argument undercuts the other, and the other does not undercut but only rebuts the first, the first defeats the second but the second does not defeat the first. We will defend this choice below when discussing Example 2.18.

Definition 2.16 *Let A_1 and A_2 be two arguments. Then A_1* defeats *A_2 iff A_1 is empty and A_2 is incoherent, or else if*

- *A_1 undercuts A_2; or*
- *A_1 rebuts A_2 and A_2 does not undercut A_1.*

We say that A_1 strictly defeats *A_2 iff A_1 defeats A_2 and A_2 does not defeat A_1.*

Corollary 2.17 *If A_1 attacks A_2, then A_1 defeats A_2 or A_2 defeats A_1. And if A_1 defeats A_2, then A_1 attacks A_2.*

To check some boundary cases, from Definition 2.14 we have that for any nonempty set of rules R, $R < \emptyset$; this makes strict arguments strictly defeat conflicting defeasible arguments. Moreover, we have that $\emptyset \not< \emptyset$, so that no strict argument strictly defeats any conflicting strict argument (recall that strict rules contain no weak literals).

We will end this section with a discussion of two main design choices. Firstly, why does undercutting defeat not depend on priorities? This is since it seems to us that the legal (and similar) conflict resolution rules are only used to resolve head-to head conflicts between rules, not to resolve head-to-body conflicts. But perhaps other domains can be found where this is different, and then we could incorporate some elements of [BRE 96] in our definitions. Brewka, who presents a prioritised version of extended logic programming that is not argument-based, reduces head-to-head conflicts to (mutual) head-to-body conflicts by letting every rule that is intended to be defeasible assume its consequent L in its antecedent as $\sim \overline{L}$. Thus some expressive power is lost, since the difference between defeasible rules like 'If A then B' and 'If A then B, unless the contrary can be shown' cannot be expressed. A more important difference is that, since in [BRE 96] *all* mutual head-to-body conflicts are resolved with priorities, this also holds for those conflicts which are not a head-to-head conflict. Consider

r_1: $\sim a \wedge \sim \neg b \Rightarrow b$
r_2: $\sim b \wedge \sim \neg a \Rightarrow a$
$r_2 < r_1$

While we regard both $[r_1]$ and $[r_2]$ as defeating each other, Brewka regards the first argument as strictly defeating the latter. A final difference with our system will be discussed in Section 9.

Our second design choice was to regard an undercutting attack as stronger than a rebutting attack. Consider first the following example.

Example 2.18 Consider

r_1 : $\sim \neg\, OJ$ is innocent $\Rightarrow OJ$ is innocent
r_2 : $\Rightarrow \neg\, OJ$ is innocent

And assume that $< \, = \emptyset$. Then, although $[r_1]$ rebuts $[r_2]$, $[r_1]$ does not defeat $[r_2]$, since $[r_2]$ undercuts $[r_1]$. So $[r_2]$ strictly defeats $[r_1]$.

With this example we can explain why we regard undercutting attack as stronger than rebutting attack. This is since the only way to accept both rules is to believe that OJ is not innocent: in that case the condition of r_1 is not satisfied. By contrast, if it is believed that OJ is innocent, then r_2 has to be rejected, in the sense that its antecedent is believed but its consequent is not. In fact, we here employ the logical counterpart of the legal principle that the law should be interpreted as coherently as possible.

If we examine the nature of our notion of defeat, we see that it is a weak one, since for rebutting an argument no $>$ relation is needed between the sets of relevant rules, but only a $\not<$ relation. So, rather than having to be 'really better' than the argument that is to be defeated, a defeating argument only has to be not inferior. The reason we have this weak notion of defeat is that our notion of 'justified arguments', to be defined next, will only regard those arguments as justified that, given the premises, are beyond any reasonable doubt or challenge. And reasonable doubt can be cast on an argument just by providing a counterargument that is not inferior to it.

For the same reason we also need the notion of strict defeat, which, being asymmetric, captures the idea of 'really being better than'. In the following section this notion will be used to ensure that a counterargument fails to cast doubt if it is inferior to at least one counterargument that is itself protected from defeat.

3 The Status of Arguments: Fixpoint Semantics

3.1 Definitions and Properties

Since defeating arguments can in turn be defeated by other arguments, comparing pairs of arguments is not sufficient; we also need to define a notion of the *status* of arguments on the basis of all ways in which they interact. In particular, the definition should allow for reinstatement of defeated arguments, if their defeater is itself (strictly) defeated by another argument. Moreover, it should respect the 'weakest link' principle that an argument cannot be justified unless all of its subarguments are justified.

This notion, then, is the central element of our system. Its definition takes as input the set of all possible arguments and their mutual relations of defeat, and produces as output a division of arguments into three classes: arguments with which a dispute can be 'won', respectively, 'lost' and arguments which leave the dispute undecided. As remarked above, the winning arguments should be only those arguments that, given the premises, are beyond reasonable doubt: the only way to cast doubt on these arguments is by providing new premises, giving rise to new, defeating counterarguments. Accordingly, we want our set of justified arguments to be unique, conflict-free, and, relative to these and some other plausible conditions, as small as possible. After defining such a set, we can define the losing, or 'overruled' arguments as those that are attacked by a justified argument and, finally, the undeciding or 'defensible' arguments as all the arguments that are neither justified nor overruled.

We will present the notion of a justified argument in two versions, which we will call a semantics and a proof-theory. With 'semantics' we here mean a specification of a set of arguments that has certain properties, and with 'proof theory' we mean a procedure for determining whether an individual argument is a member of this set. Others, e.g. [LN 95], have denoted this distinction with the terms 'declarative' and 'procedural' systems. Note that our notion of semantics leaves open that the underlying logic for constructing arguments has a model-theoretic semantics; see further Section 10.

Our semantics is based on a variant of [Dung 93b]'s notion of the acceptability of an argument with respect to a set of arguments (a detailed comparison with Dung's work will be made in Section 9).

Definition 3.1 *An argument A is* acceptable *with respect to a set $Args$ of arguments iff each argument defeating A is strictly defeated by an argument in $Args$.*

The set of justified arguments is characterised by a an operator that returns for each set of arguments the set of all arguments that are acceptable with respect to it; since this operator is monotonic, we can define the set of justified arguments as its least fixpoint. This operator is also the core of [Dung 93b]'s grounded (sceptical) semantics of extended logic programs, who calls it the 'characteristic function' of an ordered theory.

Definition 3.2 *Let Γ be an ordered theory and S any subset of $Args_\Gamma$. The* characteristic function *of Γ is:*

- $F_\Gamma : Pow(Args_\Gamma) \longrightarrow Pow(Args_\Gamma)$
- $F_\Gamma(S) = \{A \in Args_\Gamma | A \text{ } is \text{ } acceptable \text{ } with \text{ } respect \text{ } to \text{ } S\}$

Proposition 3.3 *F_Γ is monotonic.*

In this definition the notion of acceptability captures reinstatement: if A is defeated by B, it can still be justified, if and only if B is strictly defeated by an argument that is already known to be justified.

Since monotonic operators are guaranteed to have a least fixpoint, we can now define the notion of a justified argument as follows.

Definition 3.4 *For any ordered theory Γ and argument A we say that on the basis of Γ:*

1. *A is* justified *iff A is in the least fixpoint of F_Γ (denoted by $JustArgs_\Gamma$);*
2. *A is* overruled *iff A is attacked by a justified argument.*
3. *A is* defensible *iff A is neither justified nor overruled.*

And for any literal L, we say that on the basis of Γ L is a justified conclusion *iff it is a conclusion of a justified argument, L is a* defensible conclusion *iff it is not justified and it is a conclusion of some defensible argument, and L is an* overruled conclusion *iff it is not justified or defensible, and a conclusion of an overruled argument.*

The monotonicity of F_Γ gives it a constructive flavour: its least fixpoint can be approached and under a certain finiteness condition even obtained by iterative application to the empty set.

Definition 3.5 *[Dung 93b]. An ordered theory Γ is* finitary *iff each argument in $Args_\Gamma$ is attacked by at most a finite number of arguments in $Args_\Gamma$.*

Proposition 3.6 *Define for any ordered theory Γ the following sequence of subsets of $Args_\Gamma$.*

- $F^1 = F_\Gamma(\emptyset)$
- $F^{i+1} = F^i \cup F_\Gamma(F^i)$.

1. *Then $\cup_{i=0}^{\infty}(F^i) \subseteq JustArgs_\Gamma$.*
2. *If Γ is finitary, then $\cup_{i=0}^{\infty}(F^i) = JustArgs_\Gamma$.*

Below we will, unless stated otherwise, implicitly assume an arbitrary but fixed ordered theory.

In the following subsection our semantics will be illustrated with some examples. Now we state some formal properties. Firstly, by definition the set of justified arguments is unique, as we wanted. By the same definition we have that $JustArgs$ is well-behaved in the sense that all arguments in $JustArgs$ are acceptable with respect to $JustArgs$. Moreover, $JustArgs$ is in a certain sense maximal, or complete: all arguments that are acceptable with respect to $JustArgs$ are in $JustArgs$, so it contains all the arguments that a person can (on the basis of the input theory) accept as justified. But *JustArgs* is also in some sense minimal: it is the smallest set for which the previous two properties hold; so it does not contain any argument of which acceptance is not necessary: there are no circular dependencies between elements of *JustArgs*. All these properties agree with our aim to let *JustArgs* contain precisely those arguments that, given the premises, are beyond reasonable doubt.

Also our requirement that *JustArgs* is conflict-free is satisfied.

Proposition 3.7 *The set of justified arguments is conflict-free.*

Definition 3.4 does not explicitly require that all proper subarguments of a justified argument are justified, as, e.g. [Nute 94, VRE 93a, PRA 93] do. Instead, as the following proposition states, this requirement is implicit in our definitions, as also in e.g. [POL 87, SL 92, GP 92].

Proposition 3.8 *If an argument is justified, then all its subarguments are justified.*

Finally, if $Args_S$ is not conflict-free, then it is easy to check that every argument, including the empty argument, attacks itself. From this it follows that

Corollary 3.9 *For any ordered theory* $\Gamma = (S, D, <)$, *if* $Args_S$ *is not conflict-free, then every argument is defensible on the basis of* Γ.

3.2 Illustration of the Definitions

Example 3.10 The first example illustrates the step-by-step construction of Proposition 3.6.

$$\begin{array}{llll} r_0: & \Rightarrow a & r_1: & a \Rightarrow b \\ r_2: & \sim b \Rightarrow c & & \\ r_3: & \Rightarrow \neg a & & \end{array}$$

where $r_0 < r_3$. First we identify the relations of defeat. The argument $A_1 = [r_0, r_1]$ defeats the argument $A_2 = [r_2]$, since it undercuts it. Furthermore, the argument $A_3 = [r_3]$ defeats A_1, by rebutting its proper subargument $A_0 = [r_0]$ and thereby also rebutting A_1 itself.

With these relations we can construct the set of justified arguments as follows. A_3 is not defeated by any argument, since its only counterargument A_1 is too weak. So A_3 is the only argument that is acceptable with respect to $\emptyset$ and so $F^1 = \{[\,], A_3\}$ How now about A_1? It is not acceptable with respect to F^1, since it is defeated by A_3, which in turn is not strictly defeated by any argument in F^1. for the same reasons also A_0 is not acceptable with respect to F^1. How then about A_2? Although it is defeated by its only counterargument A_1, we can still add it, since A_1 is in turn strictly defeated by an argument in F^1, viz. A_3. Thus A_3 reinstates A_2 and we have that $F^2 = \{[\ ], A_3, A_2\}$. Repeating this process adds no new relevant arguments, so we can stop here: $F^3 = F^2$. Thus we obtain that A_2 and A_3 are justified, while A_0 and A_1 are overruled. Note that the weakest link principle is respected: A_1 is overruled since its subargument A_0 is overruled.

Example 3.11 We now illustrate how Definition 3.2 makes use of the notion of acceptability. Consider four arguments A, B, C, D, assume that $S = \{A, B\}$ and assume that precisely the following defeat relations hold: C and D defeat each other, A strictly defeats C and B strictly defeats D. Are C and D in $F(S)$? The answer is no: to see this in the case of C, although the argument D that defeats it, is itself strictly defeated by an argument in S, viz. B, C is also defeated by A, and A is not strictly defeated by any argument in S. So C is not acceptable with respect to S.

Example 3.12 Next we again illustrate the relation between undercutting and rebutting attack, with a variation of Example 2.18.

$r_1: \quad \Rightarrow Has\_Porsche \qquad\qquad r_2: \quad Has\_Porsche \Rightarrow Rich$

$r_3: \quad \sim Rich \Rightarrow \neg Has\_Porsche$

Assume that $r_1 < r_3$. Definition 2.16 ignores this ordering; $B = [r_3]$ does not defeat $A = [r_1, r_2]$ since A undercuts B. This causes A to be justified, as well as well as its subargument $A' = [r_1]$, although that is defeated by B. A' is reinstated since A strictly defeats its counterargument B.

Example 3.13 Condition (3) of Definition 2.2 is required because of examples like the following:

$r_1: \quad \Rightarrow a \qquad r_2: \quad a \Rightarrow a$

$r_3: \quad \Rightarrow \neg a$

Assume that $r_3 < r_2$. Then without condition (3) $[r_1, r_2]$ would be an argument saving its subargument $[r_1]$, since it defeats the argument $[r_3]$. However, with the condition $[r_1, r_2]$ is not an argument.

Example 3.14 Next we illustrate that our definitions respect the 'step-by-step' nature of argumentation: conflicts about conclusions or assumptions earlier in the chains are dealt with before 'later' conflicts.

$r_1: \ \Rightarrow a \qquad r_2: \ a \Rightarrow b \qquad r_3: \ b \Rightarrow c$

$r_4: \ \Rightarrow \neg a \qquad r_5: \ \neg a \Rightarrow d \qquad r_6: \ d \Rightarrow \neg c$

Assume that $r_4 < r_1$ and $r_3 < r_6$. Then the arguments $[r_1 - r_3]$ and $[r_4 - r_6]$ defeat each other. However, $[r_1]$ (and also $[r_1, r_2]$), strictly defeats $[r_4 - r_6]$ and has no other counterarguments; so $F^1 = \{[r_1], [r_1, r_2]\}$. But then $[r_4 - r_6]$, although defeating $[r_1 - r_3]$, is not acceptable with respect to F^2, while $[r_1 - r_3]$ is acceptable with respect to F^2, reinstated by $[r_1]$.

Example 3.15 We now illustrate the Definitions 2.13 and 2.14. The example also illustrates that conflicts that intuitively are between *sets* of arguments, can always in a suitable way be reduced to conflicts between individual arguments. Assume $S = \{s_1, s_2\}$ and $D = \{r_1, r_2, r_3\}$, where

r_1: $\Rightarrow$ *John* is taller than *Bob*
r_2: $\Rightarrow$ *Bob* is taller than *Al*
r_3: $\Rightarrow$ *Al* is taller than *John*

s_1: x is taller than $y \wedge y$ is taller than $z \Rightarrow x$ is taller than z
s_2: x is taller than $y \Rightarrow \neg\ y$ is taller than x

Because of the strict rules, at least the following three pairs of defeasible arguments attack each other.

$[r_1]$ and $[r_2, r_3]$
$[r_2]$ and $[r_1, r_3]$
$[r_3]$ and $[r_1, r_2]$

Observe that all rules of these arguments are relevant to the respective conflicts. It now depends on the priorities which arguments are justified. Assume first that $r_3 < r_1$ and $r_3 < r_2$. Then the relevant sets are ordered as follows.

$[r_1] > [r_2, r_3]$
$[r_2] > [r_1, r_3]$
$[r_3] < [r_1, r_2]$

So $[r_1]$ and $[r_2]$ are justified. How about their combination, $[r_1, r_2]$? Although $[r_1, r_2] > [r_3]$, this is not yet sufficient for concluding that also $[r_1, r_2]$ is justified; we also have to compare it with its attackers $[r_1, r_3]$ and $[r_2, r_3]$. The following relations hold:

$[r_1, r_2] > [r_1, r_3]$
$[r_1, r_2] > [r_2, r_3]$

So also $[r_1, r_2]$ is justified.

It is easy to check that if $r_3 < r_1$ and $r_2 \not< r_3$ and $r_3 \not< r_2$, then $[r_1]$ is still justified and $[r_2, r_3]$ is still overruled, but that $[r_2]$ and $[r_3]$ and also $[r_1, r_2]$ are defensible.

Example 3.16 The following example suggests that there is room for discussion whether our semantics is perhaps too sceptical.

s_1: $\rightarrow a$
r_1: $a \Rightarrow b$
s_2: $\rightarrow c$
r_2: $a \wedge c \Rightarrow \neg b$
s_3: $\rightarrow d$
r_3: $d \Rightarrow b$
r_4: $c \wedge d \Rightarrow \neg b$

Assume that the defeasible rules are ordered according to specificity: $r_1 < r_2$ and $r_3 < r_4$. Our system regards all arguments for b and $\neg b$ as defensible. Although $A_2 = [s_1, s_2, r_2]$ strictly defeats $A_1 = [s_1, r_1]$ and $A_3 = [s_2, s_3, r_4]$ strictly defeats $A_4 = [s_3, r_3]$, we also have that $[s_1, s_2, r_2]$ is defeated by $[s_3, r_3]$ and $[s_2, s_3, r_4]$ is defeated by $[s_1, r_1]$. So none of the defeasible arguments is undefeated and $F^1 = Args_S \cup \{[\,]\}$.

However, in the context of nonmonotonic inheritance [HTT 90] argue that, since all arguments for b are strictly defeated by some argument for $\neg b$, the arguments $[s_1, s_2, r_2]$ and $[s_3, r_3]$ should be justified. It would be interesting to see which change in our semantics gives this result. Note, in this respect, that $\{[s_1, s_2, r_2], [s_3, r_3]\}$ is a conflict-free fixpoint of F, albeit not the least one.

Example 3.17 Finally, we show that our system does not exhibit a form of 'extreme scepticism', to be found, for instance, in [HTT 90, Nute 94]. Informally speaking, a system is extremely sceptical if arguments have to be 'cut off' not only when they are overruled, but also when they are merely defensible. In our system this is not the case, as is illustrated by the following example.

r_1: $\Rightarrow$ f forged evidence e
r_2: f forged evidence $e \Rightarrow \neg$ e is admissible evidence
r_3: $\Rightarrow \neg$ f forged evidence e
r_4: $\sim \neg$ e is admissible evidence $\Rightarrow$ e proves guilt

Assume that $< \, = \emptyset$. According to our definitions all arguments in the example are defensible. By contrast, in extremely sceptical systems $[r_1, r_2]$ is, in our terms, not allowed to prevent $[r_4]$ from becoming justified, since it has a subargument that is only defensible. As a result, in such systems $[r_4]$ is justified, even though it is attacked by an argument that is not worse than any counterargument.

Although this difference reflects a 'clash of intuitions', we think that our intuitions are justified by our general approach. Recall that we regard an argument as justified only if, given the premises, no doubt at all can be cast on the argument. Now here doubt can be cast on the argument r_4, since it has a counterargument that is not weaker than any argument attacking it. We think that in a dispute a judge would feel compelled to determine whether $r_1 < r_3$ before deciding that e proves guilt.

4 Proof Theory with Strict Priorities

4.1 The General Idea

So far we have only provided a semantics for our system, by defining a set of arguments with the property of being justified. In this section we will define a test for membership of this set for an individual argument, i.e. we will define a proof theory for our semantics. We will make use of Proposition 3.6, which says that an argument is justified if (for finitary theories 'iff') it is in the result of finitely many iterative applications of F to the empty set. Basically, the idea is to traverse the resulting sequence $F^1, \ldots, F^n$ where A occurs for the first time in F^n, in the reverse direction. We start with A, and then for any argument B defeating A we find an argument C in F^{n-1} that strictly defeats B and so indirectly supports A. Then any argument defeating C is met with a strict defeater from F^{n-2}, and so on. Since the sequence is finite, we end with an argument indirectly supporting A that cannot be defeated.

Inspired by [VRE 93b, Dung 94, BRE 94b, LN 95] we present the proof theory in a dialectical style. A proof that an argument is justified will take the form of a dialogue tree, where each branch of the tree is a dialogue, and the root of the tree is an argument for the formula. The idea is that every move in a dialogue consists of an argument based on the input theory, where each stated argument attacks the last move of the opponent in a way that meets the player's burden of proof. That a move consists of a complete argument, means that the search for an individual argument is conducted in a 'monological' fashion, determined by the nature of the underlying logic; only the process of considering counterarguments is modelled dialectically. The required force of a move depends on who states it. Since the proponent wants a conclusion to be justified, a proponent's argument has to be strictly defeating, while since the opponent only wants to prevent the conclusion from being justified, an opponent's move may be just defeating. The reader will recognize the definition of acceptability.

Let us illustrate this with a dialogue based on the following example.

Example 4.1 Assume we have the following input theory.

r_1:	$\sim \neg\, e$ is admissible evidence $\Rightarrow e$ proves guilt
r_2:	$\Rightarrow f$ forged evidence e
r_3:	x forged evidence $y \Rightarrow \neg\, y$ is admissible evidence
r_4:	x is police officer $\wedge \sim x$ is dishonest $\Rightarrow \neg\, x$ forged evidence y
f_5:	$\rightarrow f$ is police officer in LA
f_6:	x is police officer in LA $\rightarrow x$ is police officer
r_7:	x is police officer in LA $\Rightarrow x$ is dishonest
r_8:	$\Rightarrow f$ has received a medal of honour
r_9:	x is police officer $\wedge$ x has received a medal of honour $\Rightarrow \neg\, f$ is dishonest

and assume that, for whatever reason, $r_2 < r_4$, $r_7 < r_9$ and that all other rules are incomparable. Let us denote the arguments stated by the proponent by P_i and those of the opponent by O_i. The proponent starts the dispute by asserting that $[r_1]$ is a justified argument for 'e proves guilt'.

P_1: $[r_1$: $\sim \neg\ e$ is admissible evidence $\Rightarrow e$ proves guilt$]$

Now the opponent has to defeat this argument. It can do so in only one way, by undercutting it with

O_1: $[r_2$: $\Rightarrow f$ forged evidence e,
r_3: f forged evidence $e \Rightarrow \neg\ e$ is admissible evidence$]$

The proponent now has to counterattack with an argument that strictly defeats O_1. It finds the following rebuttal, using the fact that $r_2 < r_4$.

P_2: $[f_5$: $\rightarrow f$ is police officer in LA,
f_6: f is police officer in LA $\rightarrow f$ is police officer,
r_4: f is police officer $\wedge \sim f$ is dishonest $\Rightarrow \neg\ f$ forged evidence $e]$

Again the opponent has only one way to respond, with the undercutting attack:

O_2: $[f_5$: $\rightarrow f$ is police officer in LA,
r_7: f is police officer in LA $\Rightarrow f$ is dishonest$]$

This time the proponent responds with a strictly defeating rebuttal, using that $r_7 < r_9$.

P_3: $[r_8$: $\Rightarrow f$ has received a medal of honour,
f_5: $\rightarrow f$ is police officer in LA,
f_6: f is police officer in LA $\rightarrow f$ is police officer,
r_9: x is police officer $\wedge$ x has received a medal of honour
$\Rightarrow \neg\ f$ is dishonest $]$

Now the opponent has run out of moves: no argument on the basis of our ordered theory defeats P's last argument. And since at no point could O create alternative branches, P can successfully defend its argument against every possible way of attack: r_1 is a provably justified argument.

Example 4.2 Next we again use Example 3.12, this time to illustrate how the method deals with the absolute prevalence of undercutting over rebutting defeat. To prove that John is rich, the proponent can start with

P_1: $[r_1$: $\Rightarrow$ *John* has a Porsche, r_2: *John* has a Porsche $\Rightarrow$ *John* is rich$]$

and already now the opponent has run out of moves, since

[r_3: $\sim$ *John* is rich $\Rightarrow \neg$ *John* has a Porsche]

although attacking P_1, does not defeat it, since P_1 undercuts it. Note that believing that John has a Porsche and is rich is the only way in which no rule has to be rejected; cf. Example 2.18.

In dialectical proof systems a 'loop checker' can be implemented in a very natural way, by stating that no two moves of the proponent in the same branch of the dialogue may have the same content.

Example 4.3 Consider for illustration the following familiar logic-programming example, where $S = < = \emptyset$ and $D = \{r_1, r_2\}$.

r_1: $\sim a \Rightarrow b$
r_2: $\sim b \Rightarrow a$

This is how the attempt to prove b fails.

P_1: $[r_1\colon \sim a \Rightarrow b]$
O_1: $[r_2\colon \sim b \Rightarrow a]$

Now P has run out of moves, since any defeating attack on O_1 would have the same content as P_1.

It is easy to see that a non-repetition rule will not harm P; if O had a move the first time P stated the argument, it will also have a move the second time, so no repetition by P can make P win a dialogue.

On the other hand, O must have the possibility to repeat moves, as is shown by the following example.

Example 4.4 Assume Γ is such that $S = < = \emptyset$ and $D = \{r_1, r_2, r_3, r_4\}$.

P_1: $[r_1\colon \Rightarrow \neg a]$
O_1: $[r_2\colon \sim b \Rightarrow a]$
P_2: $[r_3\colon \sim c \Rightarrow b]$
O_2: $[r_4\colon \sim a \Rightarrow c]$
P_3: $[r_2\colon \sim b \Rightarrow a]$
O_3: $[r_3\colon \sim c \Rightarrow b]$
P_4: $[r_4\colon \sim a \Rightarrow c]$

If O now cannot repeat its first move, P wins the dialogue tree. Yet $[r_1]$ is according to Definition 3.4 not justified but only defensible.

4.2 The Proof Theory

We now formally define the dialectical proof theory.

Definition 4.5 *A* dialogue *is a finite nonempty sequence of moves* $move_i = (Player_i, Arg_i)$ $(i > 0)$*, such that*

1. $Player_i = P$ *iff* i *is odd; and* $Player_i = O$ *iff* i *is even;*
2. *If* $Player_i = Player_j = P$ *and* $i \neq j$*, then* $Arg_i \neq Arg_j$*;*
3. *If* $Player_i = P$ $(i > 1)$*, then* Arg_i *is a minimal (w.r.t. set inclusion) argument strictly defeating* Arg_{i-1}*;*
4. *If* $Player_i = O$*, then* Arg_i *defeats* Arg_{i-1}*.*

The first condition says that the P begins and then the players take turns, while the second condition prevents the proponent from repeating its attacks. The remaining two conditions form the heart of the definition: they state the burdens of proof for P and O. The minimality condition on P's moves makes it impossible to make arguments trivially different by combining them with some other, irrelevant argument.

Definition 4.6 *A* dialogue tree *is a finite tree of moves such that*

1. *Each branch is a dialogue;*
2. *If* $Player_i = P$ *then the children of* $move_i$ *are all defeaters of* Arg_i*.*

The second condition of this definition makes dialogue trees candidates for being proofs: it says that the tree should consider all possible ways in which O can attack an argument of P.

Definition 4.7 *A player wins a dialogue if the other player cannot move. And a player wins a dialogue tree iff it wins all branches of the tree.*

The idea of this definition is that if P's last argument is undefeated, it reinstates all previous arguments of P that occur in the same branch of a tree, in particular the root of the tree.

Definition 4.8 *An argument* A *is* a provably justified argument *iff there is a dialogue tree with* A *as its root, and won by the proponent. And a strong literal* L *is a* provably justified conclusion *iff it is a conclusion of a provably justified argument.*

Proposition 4.9 *All provably justified arguments are justified.*

Proposition 4.10 *For finitary ordered theories each justified argument is provably justified.*

If an ordered theory is finite, then for every argument that is not justified every proof clearly fails finitely. However, this does not hold for finitary ordered theories. Here is a counterexample. Suppose Γ consists of rules r_i for every $i \in N$, defined as follows

$$r_i\colon \sim a_{i+1} \Rightarrow a_i$$

Then $JustArgs_\Gamma$ is empty, but no proof that an argument is justified fails finitely.

Finally, the weakest link principle expressed by Proposition 3.8 also holds for our proof theory.

Proposition 4.11 *If an argument is provably justified, then all its subarguments are provably justified.*

4.3 Some More Examples

Example 4.12 First we give the proof of Example 3.14.

$$\begin{array}{lllllll}
P_1\colon & [r_1\colon & \Rightarrow a, & r_2\colon & a \Rightarrow b, & r_3\colon & b \Rightarrow c] \\
O_1\colon & [r_4\colon & \Rightarrow \neg a, & r_5\colon & \neg a \Rightarrow d, & r_6\colon & d \Rightarrow \neg c] \\
P_2\colon & [r_1\colon & \Rightarrow a] & & & &
\end{array}$$

O cannot counter with $[r_4\colon\ \Rightarrow \neg a]$: that argument does not defeat P_2, since $r_4 < r_1$.

Example 4.13 Next we illustrate the non-repetition rule. Assume that $\Gamma = (S, D, <)$ where $S = \emptyset$, $D = \{r_1 - r_5\}$ as given below and $< \; = \{r_3 < r_1\}$.

$$\begin{array}{ll}
P_1\colon & [r_1\colon \Rightarrow a,\ r_2\colon\ a \Rightarrow b\ ] \\
O_1\colon & [r_3\colon \Rightarrow \neg a,\ r_4\colon\ \neg a \Rightarrow \neg b\ ] \\
P_2\colon & [r_1\colon \Rightarrow a\ ] \\
O_2\colon & [r_5\colon \sim b \Rightarrow \neg a\ ]
\end{array}$$

Now P cannot repeat P_1, but this is as it should be, since according to Definition 3.4 none of the arguments in the example are justified.

5 The formal system II: defeasible priorities

5.1 The New Fixpoint Semantics

So far we have simply assumed that there is a fixed and undisputable ordering on the rules. However, as we already said in the introduction, this assumption is often unrealistic: in legal reasoning, and also in several other domains of practical reasoning, the standards for conflict resolution are themselves subject to debate and disagreement. A full theory of defeasible argumentation should therefore also be able to formalise arguments about priorities.

Obviously, we cannot simply do this by only adding a priority predicate symbol to our object language; in addition the priorities derived at the object level somehow need to be lifted to the metatheory of the system, in particular to Definition 2.14. So we have to define a formal connection between object and metalevel.

We start by assuming that our language contains a distinguished twoplace predicate symbol $\prec$, with which information on the priorities can be expressed in the object language. This makes the third component of an ordered theory redundant, so an *ordered theory* is from now on just a pair (S, D). Next we have to make sure that the ordering thus derived is of the desired type, i.e. that it is a strict partial order. This can be done by adding the axioms of a partial order to the strict rules. A slight complication is that, since strict rules do not allow for contraposition, we also have to add the contrapositive rules for transitivity (for asymmetry the contrapositive is redundant). Thus, in the rest of this paper we assume that of every input theory the set S of strict rules contains all and only the following rules containing the $\prec$ predicate.

t_1: $x \prec y \wedge y \prec z \rightarrow x \prec z$
t_2: $x \prec y \wedge \neg\, x \prec z \rightarrow \neg\, y \prec z$
t_3: $y \prec z \wedge \neg\, x \prec z \rightarrow \neg\, x \prec y$
a: $x \prec y \rightarrow \neg\, y \prec x$

We now need to link the priority conclusions that can be drawn at the object level to the metalevel of our system: i.e. we want to make sure that $(r, r') \in <$ if and only if there is a justified argument for $r \prec r'$. In other words, the ordering component of an ordered theory Γ should now be determined by the set of all priority arguments that are justified on the basis of Γ.

To state this in terms of Proposition 3.6, our idea is that now with the set of justified arguments also the ordering is 'constructed' step-by-step (this idea is inspired by an earlier version of [BRE 96]). This means that we now have to make the dependence of the various notions on a rule ordering explicit, since this ordering now varies with the conclusions that have been drawn so far. To this end we first define the following notation, capturing what a certain set of arguments says about the priorities.

Definition 5.1 *For any set Args of arguments*

$$<_{Args} = \{r < r' \mid r \prec r' \text{ is a conclusion of some } A \in Args\}$$

And we say for any set of arguments Args that A (strictly) Args-defeats B on the basis of Γ *iff according to Definition 2.16 A (strictly) defeats B on the basis of* $(\Gamma, <_{Args})$*. Occasionally, we will also use the analogous notion Args-rebuts. And for any argument A we write* $\{A\}$*-defeats as A-defeats.*

Corollary 5.2 *If* A_1 *attacks* A_2*, then for any conflict-free set Args of arguments,* A_1 *Args-defeats* A_2 *or* A_2 *Args-defeats* A_1*; and if* A_1 *Args-defeats* A_2*, then* A_1 *attacks* A_2*.*

Next we incorporate the new notation in the definition of acceptability.

Definition 5.3 *An argument A is* acceptable *with respect to a set $Args$ of arguments iff all arguments $Args$-defeating A are strictly $Args$-defeated by some argument in $Args$.*

Thus acceptability of an argument with respect to a set $Args$ now depends on the priority conclusions of the arguments in $Args$.

In redefining the characteristic function of an ordered theory, we have to respect that in general the operator as defined above in Definition 3.4 is not monotonic. We can only prove monotonicity if the domain of the operator is restricted to conflict-free sets of arguments. Apart from this, the definition can stay the same

Definition 5.4 *Let $\Gamma = (S, D)$ be an ordered theory, S any subset of $Args_\Gamma$ and $Cargs_\Gamma$ the set of all conflict-free subsets of $Args_\Gamma$. Finally, let acceptability be defined as in Definition 5.3. Then the* characteristic function *of Γ is:*

- $G_\Gamma : Cargs_\Gamma \longrightarrow Pow(Args_\Gamma)$
- $G_\Gamma(S) = \{A \in Args_\Gamma | A \textit{ is acceptable with respect to } S\}$

Proposition 5.5 *G_Γ is monotonic.*

Definition 5.6 *For any ordered theory $\Gamma = (S, D)$ and argument A we say that A is* justified *on the basis of Γ iff A is in the least fixpoint of G_Γ (denoted by $JustArgs_\Gamma$). Overruled and defensible arguments are defined as above.*

The desirable properties that we earlier stated for fixed priorities as Proposition 3.6 als0 hold when the priorities are defeasible.

Proposition 5.7 *Define for any ordered theory Γ the following sequence of subsets of $Args_\Gamma$.*

- $G^1 = G_\Gamma(\emptyset)$
- $G^{i+1} = G^i \cup G_\Gamma(G^i)$.

1. *Then $\cup_{i=0}^{\infty}(G^i) \subseteq JustArgs_\Gamma$.*
2. *If Γ is finitary, then $\cup_{i=0}^{\infty}(G^i) = JustArgs_\Gamma$.*

Note that this time each step in the construction $JustArgs$ also implicitly extends the ordering on rules; this is because G uses the notion of acceptability, which in turn uses the notion of $Args$-defeat, and the point is that $Args$, and thus also $<_{Args}$, changes with each new step. So what happens is that at each new step the defeat relations are redetermined on the basis of the set of justified arguments that has been constructed thus far.

Proposition 5.8 *The set of justified arguments is conflict-free.*

Proposition 5.9 *If an argument is justified, then all its subarguments are justified.*

As said above, G is not monotonic if the restriction of its domain to conflict-free sets is dropped. Here is a counterexample. [7]

A_1: $[r_1\colon \Rightarrow a]$
A_2: $[r_2\colon \Rightarrow b,\ r_3\colon b \Rightarrow \neg a]$
A_3: $[r_4\colon \Rightarrow \neg b]$
A_4: $[r_5\colon \Rightarrow r_2 \prec r_4]$
A_5: $[r_6\colon \Rightarrow r_4 \prec r_2]$

$G(\{A_3, A_4\}) = \{A_1, A_3\}$
$G(\{A_3, A_4, A_5\}) = \{A_3\}$

To see whether this is harmful, we use another of Dung's notions.

Definition 5.10 *A set E of arguments is a* complete extension *iff it is a maximal (w.r.t. set inclusion) set such that $A \in E$ iff A is acceptable with respect to E.*

Dung defines this notion only for conflict-free sets E, but for our purposes it is instructive to drop this restriction. Clearly, the least fixpoint of G is a complete extension. Moreover, we have that

Proposition 5.11 *Any complete extension that is not a superset of $JustArgs$ is not conflict-free.*

Corollary 5.12 *No complete extension is a subset of $JustArgs$.*

These results mean that the restriction of the domain of G to conflict-free sets does no harm: *JustArgs* is a minimal complete extension, and all other minimal complete extensions are not conflict-free, so they do not capture a set of arguments that can all be rationally accepted as justified.

5.2 Examples

Example 5.13 We start with adding some rules on priorities to Example 3.10.

r_4: $\Rightarrow r_0 \prec r_3$
r_5: $\Rightarrow r_3 \prec r_0$
r_6: $\Rightarrow r_5 \prec r_4$

[7] In the rest of this paper we will usually leave strict arguments and combinations of independently justified arguments implicit.

The only $\emptyset$-undefeated argument is $[r_6]$, so $G^1 = \{[r_6]\}$. Then $<_{G^1} = \{r_5 < r_4\}$, so at the second step we can resolve the conflict between $[r_4]$ and $[r_5]$: $[r_4]$ strictly G^1-defeats $[r_5]$, so $G^2 = G^1 \cup \{[r_4]\}$. Now $<_{G^2} = <_{G^1} \cup \{r_0 < r_3\}$, and we can complete the example as above.

Example 5.14 We now illustrate the use of the ordering axioms in S. The example is also another illustration of the fact that a conflict that intuitively seems to be between *sets* of arguments, can always be reduced in the right way to a conflict between pairs of arguments. Assume that S only contains the priority axioms, and D contains the following rules

r_4:	$\Rightarrow r_1 \prec r_2$	r_7:	$\Rightarrow r_6 \prec r_4$
r_5:	$\Rightarrow r_2 \prec r_3$	r_8:	$\Rightarrow r_6 \prec r_5$
r_6:	$\Rightarrow r_3 \prec r_1$		

G^1 contains $[r_7]$, $[r_8]$ and their combination $[r_7, r_8]$. Then $<_{G^1} = \{r_6 < r_4, r_6 < r_5\}$, which means that $[r_4, r_5]$ strictly G^1-defeats all its counterarguments, which are $[r_6]$, $[r_4, r_6]$ and $[r_5, r_6]$. So $[r_4, r_5]$ is in G^2.

Example 5.15 The following example shows what can happen if priority examples are dependent on other arguments.

r_1: $\Rightarrow a$
r_2: $\Rightarrow \neg a$
r_3: $\Rightarrow r_2 \prec r_1$

Since $[r_3]$ is unattacked, $G^1 = \{[r_3]\}$ and then $G^2 = G^1 \cup \{[r_1]\}$. However, if we change r_3 into

r_3': $a \Rightarrow r_2 \prec r_1$

then all arguments in the example are defensible, since G^1 is now empty.

Example 5.16 Let us finally see how our definitions deal with 'pathological' cases of self-reference. Assume $D =$

r_1: $\Rightarrow r_1 \prec r_2$
r_2: $\Rightarrow r_2 \prec r_1$

Since S contains an asymmetry axiom for $\prec$, both arguments attack each other. Since $G^1 = \emptyset$, no argument in G^1 strictly G^1-defeats $[r_1]$ or $[r_2]$, so $G^2 = G^1$: neither $[r_1]$ nor $[r_2]$ is justified.

On the other hand, both arguments are defensible.

6 Proof Theory with Defeasible Priorities

6.1 Definitions

In adapting the proof theory to defeasible priorities, the main problem is on the basis of which priorities the defeating force of the moves should be determined. What we want to avoid is that we have to generate all priority arguments before we can determine the defeating force of a move. The pleasant surprise is that, to achieve this, a few very simple conditions suffice. For O it is sufficient that its move $\emptyset$-defeats P's previous move. This is so since, as shown in the appendix, Definition 2.16 implies that if A is for some S an S-defeater of P's previous move, it is also an $\emptyset$-defeater of that move. So O does not have to take priorities into account, as is illustrated by

P_1: $[r_1\colon \Rightarrow a]$

Now a possible move of O is

O_1: $[r_2\colon \Rightarrow b,\ \ r_3\colon b \Rightarrow \neg a]$

P, on the other hand, should take some priorities into account. However, it suffices to apply only those priorities that are stated by P's move; more priorities are not needed, since Definition 2.16 also implies that if P's argument Arg_i strictly $Args_i$-defeats O's previous move, it will also do so whatever more priorities will be derived. So P can reply to O_1 with

P_2: $[r_4\colon \Rightarrow \neg b,\ \ r_5\colon \Rightarrow r_2 \prec r_4]$

However, this is not the only type move that the proponent can make. If O responds with

O_2: $[r_2\colon \Rightarrow b]$

which $\emptyset$-defeats P_2, then P must have the possibility to 'undo' the defeating force of O_2 with the priority argument

P_3: $[r_5\colon \Rightarrow r_2 \prec r_4]$

We now incorporate these ideas in the definition of a dialogue. All conditions are the same as above, except for the references to 'defeat'.

Definition 6.1 *A* priority dialogue *is a finite sequence of moves* $move_i = (Player_i, Arg_i)$ $(i > 0)$, *where*

1. $Player_i = P$ *iff* i *is odd, and* $Player_i = O$ *iff* i *is even;*
2. *If* $Player_i = Player_j = P$ *and* $i \neq j$, *then* $Arg_i \neq Arg_j$*;*

3. *If $Player_i = P$ $(i > 1)$ then Arg_i is a minimal (w.r.t. set inclusion) argument such that*
    - *Arg_i strictly Arg_i-defeats Arg_{i-1}; or*
    - *Arg_{i-1} does not Arg_i-defeat A_{i-2};*
4. *If $Player_i = O$ then Arg_i $\emptyset$-defeats Arg_{i-1}.*

The only change in the definition of a dialogue tree is that the defeat condition on O's moves is made relative to the empty set.

Definition 6.2 *A* priority dialogue tree *is a finite tree of moves such that*

1. *Each branch is a dialogue;*
2. *If $Player_i = P$ then the children of $move_i$ are all $\emptyset$-defeaters of Arg_i.*

The other definitions stay the same.

Proposition 6.3 *All provably justified arguments are justified.*

Proposition 6.4 *For finite ordered theories each justified argument is provably justified.*

Proposition 6.5 *If an argument is provably justified, then all its subarguments are provably justified.*

6.2 Some Examples

Example 6.6 The first example, with the ordered theory of Example 5.15, illustrates the effect of the second option of P when responding to O.

P_1: $[r_1\colon \Rightarrow a,\ r_3\colon \Rightarrow r_2 \prec r_1]$
O_1: $[r_2\colon \Rightarrow \neg a]$
P_2: $[r_3\colon \Rightarrow r_2 \prec r_1]$

Clearly, this is a proof for a. However, if we change r_3 into

$r_3'\colon a \Rightarrow r_2 \prec r_1$

then a ceases to be provable:

P_1: $[r_1\colon \Rightarrow a,\ r_3\colon a \Rightarrow r_3 \prec r_1]$
O_1: $[r_2\colon \Rightarrow \neg a]$

Now the only way for P to undo the defeating force of O_1 is by repeating P_1, but this is not allowed. As illustrated above with Example 5.15, this agrees with the fixpoint semantics.

Example 6.7 And here is the proof that in Example 5.14 $r_1 \prec r_3$ is a justified conclusion. It branches directly after P's opening move into three dialogues. Note that in each conflict the relevant rules are all the defeasible ones occurring in the arguments.

P_1:	$[r_4, r_5]$				
O_1:	$[r_6]$	O_1:	$[r_5, r_6]$	O_1:	$[r_4, r_6]$
P_2:	$[r_7, r_8]$	P_2:	$[r_7]$	P_2:	$[r_8]$

In the remaining examples we will use the following method for naming rules. Every rule with terms $t_1, \ldots, t_n$ is named with a function expression $r(t_1, \ldots, t_n)$, where r is the informal name of the rule.

First we propose a systematic way of formalising reasoning about multiple legal orderings. The idea is to just write 'if' definitions, so that the ordering is possibly partially unspecified. The three general principles Lex Superior, Lex Specialis and Lex Posterior thus become:

$H(x, y)$: x is inferior to $y \Rightarrow x \prec y$
$T(x, y)$: x is earlier than $y \Rightarrow x \prec y$
$S(x, y)$: y is more specific than $x \Rightarrow x \prec y$

Other rules can then specify when the antecedents of these rules hold. For example,

$r_1(r_3(x), r_4(x))$: $\Rightarrow r_4(x)$ is more specific than $r_3(x)$
$r_2(x, y)$: x is in a statute $\wedge$ y is in the constitution
$\Rightarrow x$ is inferior to y

$r_1(r_3(x), r_4(x))$ says of two particular rules $r_3(x)$ and $r_4(x)$ that one is more specific than the other. This premise could be specified 'by hand' by the user but it could also be the outcome of a logical specificity definition, provided that the definition induces an ordering on individual defaults (as e.g. in [SL 92]).

Finally, the relation between the three principles can be specified by

$O_1(T(x, y), H(x, y))$: $\Rightarrow T(x, y) \prec H(x, y)$
$O_2(S(x, y), T(x, y))$: $\Rightarrow S(x, y) \prec T(x, y)$

Example 6.8 We now formalise an example in which the Italian priority rule on building regulations, mentioned in the introduction, conflicts with the temporality principle that the later rule has priority over the earlier one. They state contradicting priorities between a town planning rule saying that if a building needs restructuring, its exterior may be modified, and an earlier, and conflicting, artistic-buildings rule saying that if a building is on the list of protected buildings, its exterior may not be modified.

Note that rule r_9 states that rule r_3 is later than the Lex Posterior principle T, which implies that r_3 prevails over T, according to T itself. The application of a priority rule to itself (or better, to one of its instances), is an interesting peculiarity of this example.

$r_1(x)$:	x is a protected building $\Rightarrow \neg$ x's exterior may be modified
$r_2(x)$:	x needs restructuring $\Rightarrow$ x's exterior may be modified
$r_3(y, x)$:	x is a rule about the protection of artistic buildings $\wedge$ y is a town planning rule $\Rightarrow y \prec x$
$T(x, y)$:	x is earlier than $y \Rightarrow x \prec y$
$r_4(r_1(x))$:	$\Rightarrow r_1(x)$ is a rule about the protection of artistic buildings
$r_5(r_2(x))$:	$\Rightarrow r_2(x)$ is a town planning rule
$r_6(r_1(x), r_2(y))$:	$\Rightarrow r_1(x)$ is earlier than $r_2(y)$
$r_7(Villa_0)$:	$\Rightarrow Villa_0$ is a protected building
$r_8(Villa_0)$:	$\Rightarrow Villa_0$ needs restructuring
$r_9(T(x, y), r_3(x, y))$:	$\Rightarrow T(x, y)$ is earlier than $r_3(x, y)$

To maintain readability, we will below only give the function-symbol part of the rule names. Here is a proof that the exterior of $Villa_0$ may not be modified.

P_1: [r_7: $\Rightarrow Villa_0$ is a protected building,
r_1: $Villa_0$ is a protected building $\Rightarrow \neg$ $Villa_0$'s exterior may be modified]

O can respond in only one way.

O_1: [r_8: $\Rightarrow Villa_0$ needs restructuring,
r_2: $Villa_0$ needs restructuring $\Rightarrow$ $Villa_0$'s exterior may be modified]

P can now neutralize O's defeater with the following priority argument, saying that the protection rule prevails over the town planning rule.

P_2: [r_4: $\Rightarrow r_1$ is a rule about the protection of artistic buildings,
r_5: $\Rightarrow r_2$ is a town planning rule,
r_3: r_1 is a rule about the protection of artistic buildings $\wedge$ r_2 is a town planning rule $\Rightarrow r_2 \prec r_1$]

But O can $\emptyset$-defeat this priority argument with a conflicting priority argument based on the Lex Posterior principle.

O_2: $[r_6$: $\Rightarrow r_1$ is earlier than r_2,
T: r_1 is earlier than $r_2 \Rightarrow r_1 \prec r_2]$

Now P takes the debate to the meta-meta-level, by asserting a priority argument that neutralizes O's defeater at the meta-level. P's argument says that, since the Lex Posterior principle is earlier than the building regulations principle, the former is inferior to the latter on the basis of the former. Although this seems to be self- referential, formally it is not, since it is one instance of Lex Posterior that speaks about another instance of itself.

P_3: $[r_9$: $\Rightarrow T$ is earlier than r_3,
T': T is earlier than $r_3 \Rightarrow T \prec r_3]$

Now O has run out moves, and we know that the villa's exterior may not be modified. Note that in this dialogue the function arguments of T and r_3 are instances of r_1 and r_2, while of T' they are the full versions of T and r_3. We leave it to the computer to write the full name of T'.

7 Alternative, Credulous Semantics

Our semantics is sceptical in the sense that it defines a unique set of acceptable arguments: if two arguments are in an irresolvable conflict, neither of them is in the set. The notion of a defensible argument is, so to speak, defined on an individual basis: an argument is defensible iff it is not justified, but is neither attacked by a justified argument. However, we might want to know more than just that an individual argument is defensible. Sometimes we also want to know which defensible arguments can be defended together, i.e. which sets of defensible arguments are based on the same coherent point of view. For instance, if we have

r_1: $\Rightarrow a$
r_2: $\Rightarrow \neg a$
r_3: $\sim a \Rightarrow b$

Then we want to say that the arguments $[r_2]$ and $[r_3]$ depend on one point of view, while $[r_1]$ depends on another, incompatible point of view.

We will now investigate whether such notions are captured by an alternative semantics defined by [Dung 93b], viz. stable semantics, which has a credulous flavour. We define it for the system with defeasible priorities.

Definition 7.1 *A conflict-free set S is a* stable extension *iff every argument that is not in S, is S-defeated by some argument in S.*

Clearly, since a stable extension is conflict-free, it reflects in some sense a coherent point of view. Moreover, it is a maximal point of view, in the sense that every possible argument is either accepted or rejected.

How is the relation between stable extensions and our definition of defensible arguments? Perhaps the stable extensions are precisely the sets obtained by adding to *JustArgs* any maximal conflict-free set of defensible arguments? First we prove the following result.

Proposition 7.2 *Every stable extension includes JustArgs.*

We can now prove that if an argument is in some stable extension but not justified, it is defensible.

Proposition 7.3 *If A is in some but not all stable extensions, then A is defensible.*

Example 7.4 However, against the reverse implication there are counterexamples.

r_1: $\sim a \Rightarrow b$
r_2: $\sim b \Rightarrow c$
r_3: $\sim c \Rightarrow a$

All of $[r_1]$, $[r_2]$ and $[r_3]$ are defensible, but none of them are in a stable extension. For instance, $\{[r_1]\}$ is maximally conflict-free, but it does not defeat $[r_3]$.

Yet it seems that there is a sense in which the set $\{[r_1]\}$ reflects a maximal coherent point of view. It is captured by the following notion.

Definition 7.5 *A* defensible extension *is any maximal (w.r.t. set inclusion) conflict-free subset of Args that includes JustArgs.*

Corollary 7.6 *Every stable extension is defensible, but not every defensible extension is stable.*

However, Example 7.4 can be used that also the notion of a defensible extension is not the final answer. If we have a closer look at the defensible extension $\{[r_1]\}$, we see that $[r_1]$ is saved from being overruled by $[r_2]$, which defeats $[r_1]$'s strict defeater $[r_3]$, but the strange thing is that $[r_1]$ in turn strictly defeats its saviour $[r_2]$. So the defensible extension owes its existence to an argument which is rejected by the extension. Stable semantics avoids such situations: all the arguments that support a stable extension, are in it, since all the arguments that are not in it, are rejected by the extension.

However, this does not mean that we can equate the notion of a defensible argument with membership of a stable extension. There clearly is a difference between overruled arguments, and arguments like $[r_1]$, $[r_2]$ and $[r_3]$ in Example 7.4, which, given the input theory, are all equally good or bad. And this is the situation that is captured by the notion of a defensible extension.

With defeasible priorities defensible extensions can be nonstable for yet another reason, as can be shown by looking again at Example 5.16.

r_1: $\Rightarrow r_1 \prec r_2$
r_2: $\Rightarrow r_2 \prec r_1$

As observed earlier, both $[r_1]$ and $[r_2]$ are defensible. However, something is strange, since both of these arguments give priority to its attacker. What is the outcome of our two alternative credulous semantics?

Neither of the singleton sets are a stable extension: $[r_1]$ does not $\{[r_1]\}$-defeat $[r_2]$ and $[r_2]$ does not $\{[r_2]\}$-defeat $[r_1]$. However, both $\{[r_1]\}$ and $\{[r_2]\}$ are defensible extensions. This difference is caused by the fact that with defensible extensions only the justified priority arguments in the extension are used to determine the defeat relations; the, possibly pathological defensible priority arguments are ignored. By contrast, in checking whether a set is a stable extension, also the defensible priority arguments are used.

Concluding, we now have the following evaluative notions of arguments:

- Justified
- Defensible and stable
- Defensible but not stable
- Overruled.

8 Some Remarks on Implementation

Our proof theory already has the form of the top level of an implementation. Moreover, we have proven some results on completeness: if the input theory is finitary, i.e. it is such that each argument has at most a finite number of attackers, each justified argument has a proof that it is justified. However, it should be noted that this is a kind of completeness that does not imply semi-decidability: if the language contains function symbols, the search for arguments will not be decidable, and then the question whether an argument has counterarguments is not even semi-decidable.

In the case that an ordered theory has only a finite number of rules, decidability follows from the fact that, since arguments may not contain more than one rule with the same conclusion, a finite ordered theory has only a finite number of arguments. Note that since rules are schemes for all its instances, this condition implies that the language has no function symbols. At first sight, the exclusion of function symbols would seem problematic, since our just-explained naming convention makes heavily use of them. Yet we expect that in many practical applications also a simpler naming convention, with terms instead of function symbols, will do.

9 Related Research

9.1 The relation with Dung's approach

As for related research, we will first make a formal comparison with the work of [Dung 93b], who has developed various argumentation based semantics of extended logic programming, based on more abstract research in [BTK 93] and [Dung 95]. As already indicated above, we build on one of [Dung 93b]'s definitions, in particular on his version of well-founded semantics of extended logic programs (cf. e.g. [BS 91]). However, there are some differences. Firstly, Dung does not consider reasoning with or about priorities. Furthermore, since Dung only intends to reformulate the various semantics of logic programs, in the case of well-founded semantics his system inherits some of its features that we find less attractive. Therefore, our definitions in certain respects revise this semantics.

We will start the comparison with a formal result on the relation between Dung's formulation and our revision of well-founded semantics. To this end we assume that no priorities are defined on the rules. We first list the relevant definitions of [Dung 93b], except when his notions coincide with ours; in that case we just use our terminology.

A *d-argument* (d for 'Dung') is a set of assumptions. A *defeasible proof* is defined as our notion of an argument in Definition 2.2, but without condition (2). A defeasible proof is *based on* a d-argument A iff its assumptions are contained in A (also our arguments can be based on A in this sense). L is a *conclusion* of A iff there exists a defeasible proof for L that is based on A.

An argument A is *self-defeating* iff there exists a defeasible proof based on A for two complementary literals. An argument is *sound* iff it is not self-defeating.

Let A and B be two sound d-arguments. Then A *raa-defeats*[8] (raa for 'Reductio Ad Absurdum') B iff $A \cup B$ is self-defeating; and A *g-defeats* (g for 'ground') B iff A has for some $L \in B$ a conclusion $\overline{L}$. Finally, A *d-defeats* B iff A raa- or g-defeats B.

Observation 9.1 *Let A and B be any pair of d-arguments that are not self-defeating. Then A raa-(g-)defeats B iff some argument based on A rebuts (undercuts) B; and if an argument based on A defeats an argument based on B, then A d-defeats B.*

Finally, Dung defines acceptability as follows:

Definition 9.2 *A d-argument A is d-acceptable with respect to a set S of d-arguments iff any d-argument that d-defeats A is g-defeated by a member of S.*

[8]Here we adapt Dung's terminology to ours; recall that Dung says 'attack' where we say 'defeat'.

Note that Dung says 'g-defeated' where we say in Definition 3.1 'strictly defeated'. Without priorities this makes no difference, as the following observation states. Actually, it only makes a difference in case of defeasible priorities: then Lemma10.1(2) only holds for our definition. (Below F^d_Γ is defined as in Definition 3.2, but with as domain $Dargs_\Gamma$ instead of $Args_\Gamma$. Likewise, F^{ps}_Γ is F_Γ with domain $Args_\Gamma$.)

Observation 9.3 *Consider any input theory $\Gamma = (T, <)$.*

1. *Let S be the least fixpoint of F^d_Γ. Then all d-arguments d-defeating a member of S are strictly d-defeated by a member of S.*

2. *Let S be the least fixpoint of F^{ps}_Γ. Then all arguments defeating a member of S are strictly defeated by a member of S.*

Before we can compare Dung's grounded semantics to ours, we have to comment on what happens in case of self-defeating d-arguments. Consider the program $\Gamma = \{\Rightarrow a, \Rightarrow \neg a\}$. This program has just one argument, viz. $\emptyset$, which is self-defeating. And since Dung defines d-defeat only for sound arguments, this argument is acceptable with respect to the empty set of arguments, which makes that $\{\emptyset\}$ is the least fixpoint of F_Γ, which supports defeasible proofs for a and for $\neg a$. Dung does not comment on this possibility. We will, in agreement with earlier versions of well-founded semantics, define the set of well-founded conclusions of such programs as empty (cf. e.g. the discussion in [BRE 96, pp. 22/3]).

In the following definition $Dargs_\Gamma$ denotes for any set Γ of rules the set of all d-arguments on the basis of Γ.

Definition 9.4 *The set of* d-well-founded conclusions *of a set Γ of rules is empty if $Dargs_\Gamma$ contains a self-defeating argument; otherwise it is the least fixpoint of F^d_Γ.*

The main result of this section is that our set of justified conclusions of a program includes the d-well-founded conclusions of that program.

Proposition 9.5 *If L is a d-well-founded conclusion of Γ, then L is a justified conclusion on the basis of $(\Gamma, \emptyset)$.*

The converse does not hold. This can be shown with an example that in our view reveals a flaw of [Dung 93b]'s and [BS 91]'s well-founded semantics (similar observations have been made by [AP 95] and [BRE 96]).

Example 9.6 Consider a program with the following rules.

r_1: $\Rightarrow a$
r_2: $\Rightarrow \neg a$
r_3: $\Rightarrow b$

b is a justified conclusion of this program. However, it has only one d-argument, viz. $\emptyset$, which is self-defeating. And, as explained above, this means that the set of d-well-founded conclusions of the program is empty.

Dung's semantics can be defended by saying that it determines sets of assumptions with which a program as a whole can be given a noncontradictory interpretation. If no such set exists, the semantics 'collapses'. Our system, by contrast, still supports the derivation of conclusions from parts of the program that are not affected by the contradiction.

Finally, what is the effect of introducing priorities? It can be shown that this further enlarges the set of justified conclusions.

Proposition 9.7 *Let L be a justified conclusion on the basis of $(\Gamma, <)$. Then for any $<' \supseteq <$ it holds that L is also a justified conclusion on the basis of $(\Gamma, <')$.*

9.2 Other related research

With respect to other related research, we first discuss some earlier developments in defeasible argumentation. Pollock has in several publications, e.g. [POL 87], developed an argument-based system for defeasible reasoning, based on a sophisticated analysis of sources of defeasible rules. Pollock is mainly interested in epistemological applications of his system, and therefore puts a heavy emphasis on probability issues. In particular, he does not admit arbitrary orders on rules but only probabilities, let alone that he analyzes reasoning about priorities. For these reasons a detailed comparison is outside the scope of the present paper. Yet some remarks are in order. From a logical point of view Pollock's system has some interesting features, such as an analysis of suppositional arguments, and a distinction between conclusive and defeasible inference rules ("reasons"), with a corresponding general distinction between rebutting and undercutting defeaters. While rebutting defeaters attack the conclusion of an inference, undercutting defeaters attack the connection between premises and conclusion of an inference step. Our two forms of attack (Definition 2.8) can be seen as a special case of this distinction. In particular, attacking an assumption can be seen as denying the connection made by an application of the 'defeasible modus ponens' inference rule, which, as remarked in Section 2.2, is implicit in our notion of an argument.

Two systems that are partly inspired by Pollock's work, and that have a syntax similar to our system, are [Loui 87] and [SL 92]. What we have in common with these systems is the use of one-direction rules for representing conditionals, but the main differences are that we also have assumptions, we replace specificity with any possible source of priorities and, finally, we make the priorities defeasible.

[VRE 93a] has studied the notion of an abstract argumentation system, which has in common with [BTK 93] and [Dung 95] that it abstracts from the underlying logic for constructing arguments. Vreeswijk defines an abstract argumentation system to be a triple $(L, R, \preceq)$, where L is a logical language, R a set of inference rules and $\preceq$ a preorder on arguments. Inference rules are either strict or defeasible. Arguments can be formed by chaining inference rules into trees.

Unlike our rules, Vreeswijk's inference rules are not intended to be domain specific, but are, as in [POL 87], really inference rules, defining the logic of the language L. Also Vreeswijk's ordering component serves a different role than in our system: Vreeswijk intends it to reflect the strength of argument forms, like deduction, induction or abduction, rather than the strength of the premises used in an argument. Accordingly, Vreeswijk does not analyze reasoning about the ordering.

It seems to us that Vreeswijk's consequence notion comes close to [Dung 95]'s (credulous) preferred semantics, according to which a conflict-free set E of arguments is a preferred extension iff it is a maximal set such that each argument defeating a member of E is defeated by a member of E.

Next a comparison with default logic is in order. Syntactically our rules naturally translate into defaults: ignoring rule names, the rule $a \wedge \sim b \Rightarrow c$ becomes the default $a : \neg b / c$. Yet our semantics does not extend default logic. This can be seen most clearly with the following example (with fixed priorities), where $\Gamma = (\emptyset, \{r_1, r_2\}, \emptyset)$ and

r_1: $\Rightarrow a$
r_2: $\Rightarrow \neg a$

Then $JustArgs_\Gamma = \emptyset$, while Γ has two stable extensions, $\{[r_1]\}$ and $\{[r_2]\}$. By contrast, the default theory (F, D) with $W = \emptyset$ and $D = \{d_1, d_2\}$ as given below, has no extension.

d_1: $\top : \top / a$
d_2: $\top : \top / \neg a$

Note that we cannot translate r_1 and r_2 into the two defaults $\top : a/a$ and $\top : \neg a / \neg a$, since those defaults are the translations of $\sim \neg a \Rightarrow a$ and $\sim a \Rightarrow \neg a$.

Next we focus on some logic-programming-like systems for defeasible reasoning. One such system is [AP 95], who revise well-founded semantics for similar reasons as we discussed in Section 9.1. Although their system is not argument-based, it has two kinds of negations and prioritized rules, but not reasoning about priorities. It would be interesting to study the relation between [AP 95] and the special case of our framework with fixed priorities.

[BRE 96] develops an alternative version of extended logic programming with defeasible priorities. His version is not argument-based but instead applies a monotonic operator directly to a program, i.e. to a set of rules. Brewka defines his system on top of his own modification of [BS 91]'s formulation of well-founded semantics. In section 2.4 we have illustrated the differences with our framework in some detail. We can add that Brewka's modification of well-founded semantics partly solves the problems of this semantics, discussed in

Section 9.1, viz. if the rules in Example 9.6 are formalised as defeasible rules, i.e. if they assume their head in their body. However, if the rules are strict, the program still has no well-founded conclusions.

Also two other logic-programming-like systems are relevant, 'ordered logic' [LV 90]) and [Nute 94]'s 'defeasible logic'. Both systems have prioritized rules but not assumptions, while the priorities are fixed. They also both reflect the kind of scepticism discussed above with Example 3.17. Nute's language also has a 'might conditional', which like Pollock's undercutting defeater roughly is the conditional denial of a rule. To obtain decidability, Nute assumes that the input theory is finite and that the language has no function symbols.

Also, Nute's system has, like ours, the category of strict rules. However, as we explained in Section 2.3, his treatment of conflicts between such rules is different from ours, and in our view has some problems.

As already indicated above, our dialectical proof theory is inspired by work of several authors. In particular, in the case of fixed priorities our proof theory is a variant of [Dung 94]'s dialogue game formulation of the well-founded semantics of [Dung 93b]. Our main contribution to Dung's version is that we extend the dialectical proof theory to the case of defeasible priorities. In an epistemological context already [RES 1977] suggested the dialectical form to analyze defeasible reasoning. [BRE 94b] gives a logical reconstruction of Rescher's suggestion with the help of default logic. [VRE 93b] has developed a dialectical proof theory for his own system for defeasible argumentation developed in [VRE 93a]. Finally, Loui has in various publications defended a dialectical approach to the formalisation of defeasible reasoning, e.g. in [Loui 93].

There also is an alternative approach to reasoning about priorities, which is especially suitable for extension- or model-based systems. It was developed by [BRE 94a] for default logic, and generalised and slightly revised by [PRA 95] for any extension- or model-based system. The idea of this method is to define a criterion which a maximal set of conclusions (i.e. an extension) has to satisfy in order to count as a set of priority-based conclusions. The criterion is that what the extension 'says' about the priorities should correspond to the ordering on which the extension is based. Thus the criterion acts as a filter over the set of all possible extensions of a given theory, for any ordering. It is easy to verify that in our system every set of justified arguments satisfies the criterion.

Also two attempts to extend prioritised circumscription to reasoning about priorities should be mentioned. In [LIF 88] reasoning about priorities is still monotonic, but in [GRO 93] they can also be derived defeasibly. However, priority defaults can only solve conflicts between defaults of lower priority. In our view this makes the formalisation of some realistic legal examples impossible.

10 Open problems

Our current investigations had a specific aim: we wanted to develop a knowledge representation and reasoning formalism with certain specific features, directly useful for legal applications of AI. Although the formalism is inspired by other work, we still have glossed over a number of general logical issues. Here we briefly discuss some of them, as suggestions for further research.

Firstly, dialectical proof theories could be developed for the alternative semantics defined by [Dung 95] and for the prioritised variants of some of them defined above in Section 7 Furthermore, it is interesting to see in more detail where we deviate from, add to or specialise more abstract accounts of defeasible argumentation of e.g. [POL 87, VRE 93a, BTK 93, Dung 95].

Another interesting topic is the relation with model-theoretic approaches to the semantics of extended logic programming, in particular with partial semantics; see e.g. [WAG 94]. Briefly, the general idea of this semantics is that an interpretation I is not just one set of atoms (supposed to be true) but two sets, I^+ and I^-, where I^+ contains the atoms that are definitely true and I^- contains the atoms that are definitely false. Let Γ be an extended logic program. Then for any atom P, $\Gamma \models \sim P$ iff $P \notin I^+$, while $\Gamma \models \neg P$ iff $P \in I^-$. On top of this, various semantic consequence notions can be defined, but if they respect the intended reading of weak negation as negation as failure, they will clearly be nonmonotonic, being a preferred model semantics.

The following research issues suggest themselves. Firstly, for undercutting and rebutting attack the following semantic constraints might be defined (assuming there are no priorities). If A is an argument with conclusion L, then B undercuts A iff $A \cup B \not\models L$, while A rebuts B iff $A \cup B \models \bot$. Moreover, it would be interesting to see whether our view on the relation between rebutting and undercutting defeat is reflected by the semantics.

The priorities come in as follows. Our present framework would partly be an alternative formulation of the partial semantics, viz. an argument-based analysis of the nonmonotonicity arising from weak negation, but it would also add something on top of the semantics, viz. a method for inconsistency handling for programs that have no models (cf. our last paragraph on Dung in Section 9). This is where the priorities are relevant.

Another semantic constraint on our framework could be that if φ is a justified conclusion on the basis of Γ, and $\Gamma' \subseteq \Gamma$ is the set of all rules occurring in all arguments that are justified on the basis of Γ, then $\Gamma' \models \varphi$; and if Γ has a model then $\Gamma \models \varphi$ iff φ is justified on the basis of Γ.

Thus the partial semantics could be used as support for certain features of our framework. On the other hand, the argument-based approach could also support elements of the semantics: since our treatment of negation as failure in terms of undercutting attack (which is based on [BTK 93] and [Dung 93b]) seems quite natural, it might serve as a criterion for choosing between alternative semantic treatments.

Conclusion

In this paper we have made several contributions to the logical study of defeasible reasoning with prioritised rules. Our main contribution has been a formalisation of reasoning about priorities, in both a semantical and a prooftheoretical form. Two other additions to the literature are a detailed study of reasoning with prioritised rules that also contain exception clauses, and an alternative to [Nute 94]'s treatment of strict one-direction rules. To obtain representations of knowledge that respect the surface structure of natural language, we have developed our system in the form of an argument-based system.

On some points alternative definitions may be possible; for instance, examples might be found which motivate a different relation between rebutting and undercutting arguments. However, we think that our system is robust enough to allow for such adjustments without having to change its overall structure.

Finally, although with an eye to implementation we have based our system on a logic-programming like language, the relevance of our system is not restricted to logic-programming; since our system is essentially an instance of the abstract theory of [BTK 93] and [Dung 95], its general form can be applied to any underlying logical language.

Acknowledgments

The authors thank Mark Ryan for his comments on an earlier version of this paper, and the SG1 Working Group of ModelAge, led by Pierre-Yves Schobbens, for useful discussions.

Appendix: Proofs

For the system with fixed priorities some results are below not explicitly proven; their proofs can easily be obtained from the corresponding results for defeasible priorities, by omitting the applications of the Lemma's 10.1 and 10.3.
The following two lemmas are crucial in proving the results for the system with defeasible priorities.

Lemma 10.1 *For any two conflict-free sets of arguments S and S' such that $S \subseteq S'$, and any two arguments A and B we have that*

1. *If A S'-defeats B, then A S-defeats B.*
2. *If A strictly S-defeats B, then A strictly S'-defeats B.*

Proof: Observe first that undercutting is independent of S and S'. Then (1) follows since if no argument in S' has a conclusion $r \prec r'$, also no argument in S has that conclusion. For (2) observe first that if S contains an argument A with conclusion $r \prec r'$, also S' contains A and by conflict-freeness S' does not contain an argument for $r' \prec r$; so for any B and C, if B strictly S-defeats C, then B S'-defeats C. Moreover, by contraposition of (1) C does not S'-defeat B if C does not S-defeat B, so C strictly S'-defeats B. ■

Lemma 10.2 *If an argument is according to Definition 5.3 acceptable with respect to a conflict-free set $Args$, it is also acceptable with respect to any conflict-free superset of $Args$.*

Proof: Assume $Args_1 \subseteq Args_2$ A is acceptable wrt $Args_1$ and B $Args_2$–defeats A. Then by Lemma 10.1 B also $Args_1$–defeats A, and then there is a $C \in Args_1$ that strictly $Args_1$- defeats B. But then by Lemma 10.1 C also strictly $Args_2$-defeats B and since $C \in Args_2$, A is acceptable with respect to $Args_2$. ■

Proposition 3.3 *F_Γ is monotonic.*

Proof: From the fact that the proof of Lemma 10.2 clearly also applies to Definition 3.1. ■

Proposition 3.7 *The set of justified arguments is conflict-free.*

Proof: Assume S is not conflict-free and S is the least fixpoint of F. Let S^c be the set of all arguments in conflict with any argument in S, and let S^- be S/S^c. We prove that $F(S^-) = S^-$.

Assume for contradiction the contrary. Since F is monotonic, $F(S^-) \subseteq S$. But since no argument in S^- attacks any argument in S^c, no argument in S^c is acceptable with respect to S^-, so $F(S^-) \subset S^-$.

Then some $A \in S^-$ is not acceptable with respect to S^-, so there is a B that defeats A and that is not strictly defeated by any $C \in S^-$. Then B is acceptable with respect to S^-, so $B \in F(S^-)$. Then since F is monotonic, also $B \in S$. But by choice of S^- $B \notin S$. ■

Proposition 5.5 *G_Γ is monotonic.*

Proof: This follows immediately from Lemma 10.2. ■

Proposition 5.7 *Define for any ordered theory Γ the following sequence of subsets of $Args_\Gamma$.*

- $G^1 = G_\Gamma(\emptyset)$
- $G^{i+1} = G^i \cup G_\Gamma(G^i)$.

1. *Then $\cup_{i=0}^{\infty}(G^i) \subseteq JustArgs_\Gamma$.*
2. *If Γ is finitary, then $\cup_{i=0}^{\infty}(G^i) = JustArgs_\Gamma$.*

Proof:

1. Immediate from monotonicity of G.
2. A special case of Lemma 11 of [Dung 93b].

■

Proposition 5.8 *The set of justified arguments is conflict-free.*

Proof: This follows if we can prove that if $Args$ is conflict-free, also $G(Args)$ is conflict-free. Assume $Args$ is conflict–free, $A, A' \in G(Args)$ and A attacks A'. Then by Corollary 5.2 A $Args$–defeats A' or A' $Args$–defeats A. Assume without loss of generality that A $Args$-defeats A'. Then, since $A' \in G(Args)$, some $B \in Args$ strictly $Args$-defeats A. But then, since $A \in G(Args)$, some $B' \in Args$ strictly $Args$-defeats B. But then $Args$ is not conflict-free. ■

Proposition 5.9 *If an argument is justified, then all its subarguments are justified.*

Proof: Suppose $A \in JustArgs$, A' is a subargument of A and B $JustArgs$-defeats A'. Then two cases have to be considered. Firstly, if B does not $JustArgs$-defeat A, this is because A undercuts B. But then A strictly $JustArgs$-defeats B. Secondly, if B $JustArgs$-defeats A, then some $C \in JustArgs$ strictly $JustArgs$-defeats B. In both cases A' is acceptable with respect to $JustArgs$ and so $A' \in JustArgs$. ■

Proposition 5.11 *Any complete extension that is not a superset of $JustArgs$ is not conflict-free.*

Proof: This follows since $JustArgs$ is the smallest conflict-free fixpoint of G. ■

Proposition 6.3 *All provably justified arguments are justified.*

Proof: Consider a proof for an argument A and consider first the leaf Arg_n of any longest branch of the proof. Note that $Player_n = P$. We prove by induction that all of P's arguments in the proof are in $JustArgs$. Note first that since O cannot move at $n + 1$, Arg_n has no $\emptyset$-defeaters. But then by Lemma 10.1 Arg_n has no $JustArgs$-defeaters and so $Arg_n \in JustArgs$.

Consider next any Arg_i such that $Arg_{i+2} \in JustArgs$ and observe that all Arg_{i+1} that $\emptyset$-defeat Arg_i are strictly Arg_{i+2}-defeated by Arg_{i+2} or Arg_i. But then by Lemma 10.1(2) they are also strictly $JustArgs$-defeated by Arg_{i+2} or Arg_i. But then all arguments that $\emptyset$-defeat Arg_i are strictly $JustArgs$-defeated by Arg_{i+2}. Among those are all arguments that $JustArgs$-defeat Arg_i. But then $Arg_i \in JustArgs$. Since this in particular holds for Arg_1 we have that $Arg_1 \in JustArgs$. ■

Proposition 6.5 *If an argument is provably justified, then all its subarguments are provably justified.*

Proof: Assume A is provably justified, D is a proof for P, and A' is a subargument of A. We construct a proof for A' as follows. First we prune from D

all branches starting with children of A that do not $\emptyset$-defeat A'. Clearly every branch of the resulting tree D' is won by P. Next we add in D' to the children of A' all arguments B that $\emptyset$-defeat A' but that do not $\emptyset$-defeat A and thus do not occur in D. Then for all such B by Definition 2.16 and Corollary 5.2 A strictly $\emptyset$-defeats B and by Lemma 10.1 A strictly A-defeats B. Recall that A has a proof, which is D. Now if A' does not occur in D, D' can after B be extended with D. Clearly the result is then a proof for A'. This is not possible, however, if A' occurs in D, since this would violate the non-repetition rule. Still, because of the same rule D is in this case after B extended with an argument $C \neq A$ such that C strictly C-defeats B. And then D' can after B be extended with the subtree D'' of D with root C. And since by non-repetition A' does not occur in D'', the result is again a dialogue tree for A' won by P. ■

To prove completeness, we first prove the following lemma.

Lemma 10.3 *For any conflict-free set of arguments S and arguments A and B: if A strictly S-defeats B, there is a minimal (w.r.t. set inclusion) $X \subseteq S$ such that A strictly X-defeats and strictly $A+X$-defeats B.*

Proof: Since Definition 2.16 only considers a finite number of priority relations, A strictly X-defeats B for some finite and $\subseteq$-minimal $X \subseteq S$. Then by Lemma 10.1(2) A also strictly $A+S$-defeats B. ■

Proposition 6.4 *For finite ordered theories each justified argument is provably justified.*

Proof: Assume $Arg_1 \in JustArgs$. Then there is a finite sequence of conflict-free sets $G^0, \ldots, G^n$ such that, firstly, $G^0 \cup \ldots \cup G^n \in JustArgs$ and, secondly, there is a smallest n such that $Arg_1 \in G^n$.

Next we construct a tree of arguments as follows. Its root is Arg_1, and its children are all its $\emptyset$-defeaters Arg_2. We prove that if there are such children, then P can reply with an argument from G^{n-1}. Consider any child Arg_2 of Arg_1. If Arg_2 does not G^{n-1}-defeat Arg_1, then by Lemma 10.3 there is a $\subseteq$-minimal argument $X \in G^{n-1}$ such that Arg_2 does not X-defeat Arg_1. But then P can move with X.

So Arg_2 G^{n-1}-defeats Arg_1. We have to prove that there is an argument $B \in G^{n-1}$ that strictly B-defeats Arg_2. Since $Arg_1 \in G^n$, we know that there is a $B \in G^{n-1}$ that strictly G^{n-1}-defeats Arg_2, and so by Lemma 10.3 that for some finite $X \subseteq G^{n-1}$ B also strictly $B+X$-defeats Arg_2. We now have to prove that also $B+X$ strictly $B+X$-defeats Arg_1. The idea is that this holds if Arg_1 does not attack X.

To find such a B and X, consider the largest $G^j \subset G^n$ that contains no arguments in conflict with Arg_2. Then some of them are in G^{j+1}. Consider any $\subseteq$-minimal such argument B. Then, since Arg_2 has no attackers in G^j, B can only be in G^{j+1} since B strictly G^j-defeats Arg_2. But then by Lemma 10.3

there is a $\subseteq$-minimal $X \in G^j$ such that B strictly $B+X$-defeats Arg_2. Then, since by choice of G^j Arg_2 does not attack X, clearly also $B+X$ is a $\subseteq$-minimal argument strictly $B+X$-defeating Arg_2. Clearly $B+X \in G^{j+1}$, and since $j+1 \leq n-1$ and thus $G^{j+1} \subseteq G^{n-1}$, $B+X \in G^{n-1}$.

It is left to verify the non-repetition rule. By choice of G^{n-1}, $Arg_1 \neq B+X$, and by Proposition 3.8 an argument is not in G^{n-1} if not all its subarguments are. So P can move with $B+X$.

Now we repeat the process for P's move and the largest i such that its content is not in G^i. Since $i < n-1$ and $G^0, \ldots, G^n$ is finite, we need to repeat only a finite number of times. Observe that the leaves have by construction of G^1 no $\emptyset$-defeaters, so O cannot move. Then by its construction the tree is a priority dialogue tree for Arg_1 won by P. ■

Proposition 7.2 *Every stable extension includes JustArgs*

Proof: Assume E is a stable extension, $J = JustArgs$ and assume that $J/E \neq \emptyset$. Note that every argument A in J/E is E-defeated by some argument B in E. If for no such B any argument in $J \cap E$ strictly $J \cap E$-defeats B, then $F(J \cap E) = J \cap E$ and J is not the least fixpoint of G. So some such B is strictly $J \cap E$-defeated by an argument $C \in J \cap E$. But then E is not conflict-free. ■

Proposition 7.3 *If A is in some but not all stable extensions, then A is defensible.*

Proof: Immediate by conflict-freeness of stable extensions and by Proposition 7.2. ■

Observation 9.3 *Consider any input theory $\Gamma = (T, <)$.*

1. *Let S be the least fixpoint of F^d_Γ. Then all d-arguments d-defeating a member of S are strictly d-defeated by a member of S.*

2. *Let S be the least fixpoint of F^{ps}_Γ. Then all arguments defeating a member of S are strictly defeated by a member of S.*

Proof: We prove it for (1); the proof of (2) is similar. Assume for contradiction that $A \in S$, B d-defeats A and no $C \in S$ strictly d-defeats B. Then let G be the set of all arguments in S that g-defeat B. Then B g-defeats all members of G and so $S - G$ is a fixpoint of F^d. ■

Proposition 9.5 *If L is a d-well-founded conclusion of Γ, then L is a justified conclusion on the basis of $(\Gamma, \emptyset)$.*

Proof: Note first that the proposition trivially holds if $Dargs_\Gamma$ contains a self-defeating d-argument. For the other case we will prove that without priorities the two semantics coincide.

As for notation, S^{ps} denotes for any set S of d-arguments the set of all arguments based on any member of S; reversely for S^d. And for any d-argument A^d, A^{ps} denotes an argument based on A^d; conversely, for any argument A^{ps}, A^d denotes the corresponding d-argument, i.e. the set of all assumptions of A^{ps}.

Lemma 10.4 *Let Γ be any set of rules such that $Dargs_\Gamma$ does not contain self-defeating arguments. Then the least fixpoint of F^d_Γ is the least fixpoint of F^{ps}_Γ.*

Proof: The proof needs the following lemma.

Lemma 10.5 *Consider any set S^{ps} of arguments. For any A^{ps} that is acceptable wrt S^{ps}, A^d is acceptable wrt S^d.*

Proof: Assume A^{ps} is acceptable wrt S^{ps}. Then if B^{ps} defeats A^{ps}, some $C^{ps} \in S^{ps}$ strictly defeats B^{ps}. Then by Observation 9.1 C^d g-defeats B^d, so A^d is acceptable wrt S^d. ■

Next we prove two results from which Lemma 10.4 follows immediately.

1. If S^d is the least fixpoint of F^d, then S^{ps} is a fixpoint of F^{ps}.

Proof: First we show that S^{ps} contains all arguments that are acceptable to it. By Lemma 10.5 for any A^{ps} that is acceptable wrt S^{ps}, A^d is acceptable wrt S^d. So, since $A^d \in S^d$, $A^{ps} \in S^{ps}$.

Secondly, we show that all members of S^{ps} are acceptable to it. Consider any $A^d \in S^d$. We want to show that all A^{ps} are acceptable wrt A^{ps}. If B^d d-defeats A^d, then by Observation 9.3 some $C^d \in S^d$ strictly d-defeats B^d. Then by Observation 9.1 any B^{ps} defeating some A^{ps} is strictly defeated by some C^{ps}. So, since $C^{ps} \in S^{ps}$ (because $C^d \in S^d$), all such A^{ps} are acceptable wrt S^{ps}. ■

2. If S^{ps} is the least fixpoint of F^{ps}, then S^d is a fixpoint of F^d.

Proof: First we show that S^d contains all arguments that are acceptable to it. Assume A^d is acceptable wrt S^d. Then if B^d d-defeats A^d, some $C^d \in S^d$ g-defeats B^d. Then by Observation 9.1 any B^{ps} defeating some A^{ps} is undercut by some $C^{ps} \in S^{ps}$. Moreover, by Lemma 9.3(2) some such C^{ps} strictly defeats B^{ps}. So, since $C^{ps} \in S^{ps}$, all such A^{ps} are acceptable wrt S^{ps}. But then they are in S^{ps} and so $A^d \in S^d$.

Secondly, we want to show that all members of S^d are acceptable to it. Assume $A^d \in S^d$. Then all A^{ps} are acceptable wrt S^{ps}. Then by Lemma 10.5 A^d is acceptable to A^d. ■
From (1) and (2) it immediately follows that the least fixpoint of F^d is the least fixpoint of F^{ps}. ■

■

Proposition 9.7 *Let L be a justified conclusion on the basis of* $(\Gamma, <)$. *Then for any* $<' \supseteq <$ *it holds that L is also a justified conclusion on the basis of* $(\Gamma, <')$.

Proof: Immediate from Lemma 10.1. ■

References

[AP 95] A. Analyti and S. Pramanik. Reliable semantics for extended logic programs with rule prioritization. *Journal of Logic and Computation* 5 (1995), 303–324.

[BS 91] C. Baral and V. Subrahmanian, Duality between alternative semantics of logic programs and nonmonotonic formalisms. *Proceedings of the International Workshop on Logic Programming and Nonmonotonic Reasoning*, Springer Verlag, Berlin, 1991.

[BTK 93] A. Bondarenko, F. Toni, R.A. Kowalski. An assumption-based framework for non-monotonic reasoning. *Proceedings of the Second International Workshop on Logic Programming and Nonmonotonic Reasoning*, MIT Press, 1993, 171–189.

[BRE 94a] G. Brewka. Reasoning about priorities in default logic. *Proceedings AAAI-94*, 247–260.

[BRE 94b] G. Brewka. A logical reconstruction of Rescher's theory of formal disputation based on default logic. *Proceedings of the 11th European Conference on Artificial Intelligence*, 366–370.

[BRE 96] G. Brewka. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research* 4 (1996), 19–36.

[Dung 93b] P.M. Dung. An argumentation semantics for logic programming with explicit negation. *Proceedings of the Tenth Logic Programming Conference*, MIT Press 1993, 616–630.

[Dung 94] P.M. Dung. Logic programming as dialogue games. Unpublished paper.

[Dung 95] P.M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n–person games. *Artificial Intelligence* 77 (1995), 321–357.

[GP 92] H. Geffner and J. Pearl. Conditional entailment: bridging two approaches to default reasoning. *Artificial Intelligence* 53 (1992), 209–244.

[GOR 95] T.F. Gordon. *The Pleadings Game. An Artificial Intelligence Model of Procedural Justice.* Kluwer, Dordrecht, 1995.

[GRO 93] B.N. Grosof. Prioritizing multiple, contradictory sources in commonsense learning by being told; or, advice-taker meets bureaucracy. *Proceedings Common Sense '93: The second Symposium on Logical Formalizations of Common-Sense Reasoning*, Austin, Texas, 1993.

[HTT 90] J.F. Horty, R.H. Thomasson and D.S. Touretzky. A sceptical theory of inheritance in nonmonotonic semantic networks. *Artificial Intelligence* 42 (1990), 311–348.

[LV 90] E. Laenens and D. Vermeir. A fixed points semantics for ordered logic. *Journal of Logic and Computation* Vol. 1 No. 2, 1990, 159–185.

[LIF 88] V.L. Lifschitz. Circumscriptive theories: a logic-based framework for knowledge representation. *Journal of Philosophical Logic* 17 (1988), 391–441.

[Loui 87] R.P. Loui. Defeat among arguments: a system of defeasible inference. *Computational Intelligence* 2 (1987), 100–106.

[Loui 93] R.P. Loui. Process and policy: resource-bounded non-demonstrative reasoning. Report WUCS–92–43, Washington–University–in–St–Louis, 1993. To appear in *Computational Intelligence*.

[LN 95] R.P. Loui and J. Norman. Rationales and argument moves. *Artificial Intelligence and Law* 3: 159–189, 1995.

[LNOM 93] R.P Loui, J. Norman, J. Olson, and A. Merrill. A design for reasoning with policies, precedent, and rationales. *Proceedings of Fourth International Conference on Artificial Intelligence and Law*, ACM Press, 1993, 202–211.

[Nute 88] D. Nute. Defeasible reasoning and decision support systems. *Decision Support Systems* 4 (1988), 97–110.

[Nute 94] D. Nute. A decidable quantified defeasible logic. In D. Prawitz, B. Skyrms and D. Westerståhl (eds.): *Logic, Methodology and Philosophy of Science IX.* Elsevier Science B.V., 1994, 263–284.

[POL 87] J.L. Pollock. Defeasible reasoning. *Cognitive Science* 11 (1987), 481–518.

[PRA 91] H.Prakken. A tool in modelling disagreement in law: preferring the most specific argument. *Proceedings of the Third International Conference on Artificial Intelligence and Law*, Oxford 1991. ACM Press 1991, 165–174.

[PRA 93] H. Prakken. An argumentation framework in default logic. *Annals of Mathematics and Artificial Intelligence*, 9 (1993) 91–131.

[PRA 95] H. Prakken. A semantic view on reasoning about priorities (extended abstract). *Proceedings of the Second Dutch/German Workshop on Nonmonotonic Reasoning*, Utrecht 1995, 152–159.

[PS 96a] H. Prakken and G. Sartor. A system for defeasible argumentation, with defeasible priorities. *Proceedings of the International Conference on Formal Aspects of Practical Reasoning (FAPR'96)*, Bonn 1996. Springer Lecture Notes in AI 1085, Springer Verlag, 1996, 510–524.

[PS 96b] H. Prakken and G. Sartor. A dialectical model of assessing conflicting arguments in legal reasoning. To appear in *Artificial Intelligence and Law.*

[RES 1977] N. Rescher. *Dialectics: a controversy-oriented approach to the theory of knowledge.* State University of New York Press, Albany, 1977.

[SAR 94] G. Sartor. A formal model of legal argumentation. *Ratio Juris* 7 (1994), 212–226.

[SL 92] G.R. Simari and R.P. Loui. A mathematical treatment of defeasible argumentation and its implementation. *Artificial Intelligence* 53 (1992), 125–157.

[VRE 93a] G. Vreeswijk. *Studies in defeasible argumentation.* Doctoral dissertation Free University Amsterdam, 1993.

[VRE 93b] G. Vreeswijk. Defeasible dialectics: a controversy-oriented approach towards defeasible argumentation. *Journal of Logic and Computation*, 1993, Vol. 3, No. 3, 317–334.

[WAG 94] G. Wagner. *Vivid logic - Knowledge-based reasoning with two kinds of negation.* Springer Lecture Notes on AI 764, Springer Verlag, Berlin.