An Inclusion Theorem for Defeasible Logics

DAVID BILLINGTON
School of ICT, Griffith University
GRIGORIS ANTONIOU
Institute of Computer Science, FORTH &
Computer Science Department, University of Crete, Greece
GUIDO GOVERNATORI
NICTA
and
MICHAEL MAHER
NICTA & University of New South Wales

Defeasible reasoning is a computationally simple nonmonotonic reasoning approach that has attracted significant theoretical and practical attention. It comprises a family of logics that capture different intuitions, among them ambiguity propagation versus ambiguity blocking, and the adoption or rejection of team defeat. This paper provides a compact presentation of the defeasible logic variants, and derives an Inclusion Theorem which shows that different notions of provability in defeasible logic form a chain of levels of proof.

Categories and Subject Descriptors: I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; F4.1 [**Mathematical Logic and Formal Languages**]: Computational Logic

General Terms: Theory

Additional Key Words and Phrases: Ambiguity Blocking, Ambiguity Propagation, Defeasible Logic, Defeasible Logic Variants, Team Defeat

1. INTRODUCTION

Nonmonotonic reasoning provides techniques for reasoning with incomplete information. Typically this is done by "filling the gaps" in the available certain informa-

The work was partially supported by the Australian Research Council.
NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.
Authors' addresses: D. Billington, School of Information and Communication Technology, Griffith University, Nathan, QLD 4111, Australia, email: d.billington@griffith.edu.au; G. Antoniou, Institute of Computer Science, FORTH, P.O. Box 1385, 71110 Heraklion Crete. Greece, email: antoniou@ics.forth.gr; G. Governatori, NICTA, PO Box 6020, St Lucia, QLD 4067, Australia, email: guido.governatori@nicta.com.au. M.J. Maher, NICTA, Locked Bag 6016, University of New South Wales, Sydney NSW 1466, Australia, email: michael.maher@nicta.com.au.

tion by making some kind of plausible (or desirable) assumptions. Classical nonmonotonic reasoning methods include default logic [Reiter 1980], circumscription [McCarthy 1980], autoepistemic logic [Moore 1985] and nonmonotonic inference relations [Kraus et al. 1990; Makinson 1994].

Nute proposed a simple, rule-based nonmonotonic approach, *defeasible logic* [Nute 1994], where a conclusion supported by a rule might be overturned by the effect of another rule. Roughly, a proposition p can be defeasibly proved only when a rule supports it, and it has been demonstrated that no applicable rule supports $\neg p$. These logics also have a monotonic reasoning component, and a priority relation on rules to resolve certain conflicts among rules. One advantage of these logics is that the cost of computing with them is low [Maher 2001], in contrast to most nonmonotonic logics.

Defeasible logics have recently attracted considerable interest. Theoretically, they have been studied deeply in terms of proof theory [Antoniou et al. 2001], semantics [Governatori et al. 2004; Maher 2002], and its relation to logic programming [Antoniou et al. 2006; Antoniou et al. 2000]. In addition, the use of defeasible logics and related approaches in various application domains has been advocated, including the modelling of regulations and business rules [Grosof et al. 1999], modelling of contracts [Reeves et al. 2002; Governatori 2005], legal reasoning [Prakken 1997; Governatori et al. 2005], agent negotiations [Dumas et al. 2002] and agent modelling [Governatori et al. 2006; Governatori and Rotolo 2008]. Also, rule systems based on defeasible logic play an important role in the development of the rule layer of the semantic web [Bassiliades et al. 2006; Antoniou and Bikakis 2007].

In monotonic logics, there is generally a consensus about what constitutes a semantic consequence. Thus, monotonic reasoning systems are compared in terms of their respective expressive power and computational complexity. In contrast, nonmonotonic logics seek to fill gaps in the certain knowledge, and there are various, often clashing intuitions about what constitutes a sensible consequence, even on the basis of the same knowledge base. As an instantiation of this general phenomenon, various notions of provability have been proposed for defeasible reasoning, capturing various levels of proof (definite and defeasible provability), as well as different intuitions. The latter include the propagation or blocking of ambiguity, and the adoption or rejection of team defeat. Although we focus on defeasible logics in this paper, these issues play a role in broader terms in the field of nonmonotonic reasoning [Horty 2001; 2002].

At the technical level, the main contribution of this paper is a comparative study of the relative strength of different notions of inference within the framework of defeasible logics. But based on the previous discussion, this work can also be regarded as indicative of the way nonmonotonic systems can be compared, and is thus of more general interest. For example, it has been stated in the literature that ambiguity propagating behaviour results in fewer conclusions being drawn. Our main result demonstrates this formally in the context of defeasible logics.

The paper is organised as follows. We will first introduce an abstract modular construction of the proof theory of defeasible logic in Section 2. This construction will be further explained in Section 3, where concrete instances of variants of Defeasible Logic will be defined to capture various intuitions. Section 4 contains a

number of properties shared by the defeasible logic variants. The main technical result of this paper is an Inclusion Theorem which shows that the different types of provability introduced form a chain of increasing strength. This result is presented and discussed in Section 5 and its proof is found in an Appendix.

Throughout this paper, we focus on finite propositional theories, as is common in non-monotonic reasoning. A predicate form of defeasible logics is outlined in [Maher and Governatori 1999], but to treat such theories requires complications of the proof theory that distract from the inclusion properties we wish to investigate. Furthermore, by restricting our attention to finite theories we are able to give constructive proofs of the inclusions.

2. ABSTRACT DEFEASIBLE LOGICS

2.1 Syntax

A *defeasible theory* D is a triple $(F, R, >)$ where F is a finite set of literals (called *facts*), R a finite set of rules, and $>$ a superiority relation on R. In expressing the proof theory we consider only propositional rules. Rules containing free variables are interpreted as the set of their variable-free instances (with respect to a finite Herbrand universe).

Formally, a *rule* has three parts: a finite set (possibly empty) of literals on the left, an arrow in the middle, and a literal on the right. The set of literals on the left of the arrow of a rule r is called the *antecedent* of the rule, $A(r)$, and the literal on the right of the arrow is called the *consequent* of the rule, $c(r)$. When the antecedent of a rule is just a singleton set, say $\{a\}$, we usually omit the set braces around a. The set of consequents of a set of rules S is denote by $c(S) = \{c(r) : r \in S\}$.

There are three kinds of rules: Rules which contain the *strict arrow* $\rightarrow$ are called *strict rules* and are interpreted in the classical sense: whenever the premises are indisputable (e.g. facts) then so is the conclusion. An example of a strict rule is "Emus are birds". Written formally: $emu(X) \rightarrow bird(X)$.

Rules which contain the plausible arrow $\Rightarrow$ are called *defeasible rules*, and can be defeated by contrary evidence. An example of such a rule is $bird(X) \Rightarrow flies(X)$ which reads as follows: "Birds typically fly".

Rules which contain the arrow $\rightsquigarrow$ are called *defeaters*, and are used to prevent some conclusions. In other words, they are used to defeat some defeasible rules by producing evidence to the contrary. An example is the rule $heavy(X) \rightsquigarrow \neg flies(X)$ which reads as follows: "If an animal is heavy then it may not be able to fly". The main point is that the information that an animal is heavy is not sufficient evidence to conclude that it doesn't fly. It is only evidence that the animal *may* not be able to fly.

Given a set R of rules, we denote the set of all strict rules in R by R_s, the set of strict and defeasible rules in R by R_{sd}, the set of defeasible rules in R by R_d, and the set of defeaters in R by R_{dft}. $R[q]$ denotes the set of rules in R with consequent q.

A *superiority relation* on R is an acyclic relation $>$ on R (that is, the transitive closure of $>$ is irreflexive). When $r_1 > r_2$, then r_1 is called *superior* to r_2, and r_2 *inferior* to r_1. This expresses that r_1 may override r_2.

When $>$ is applied to rules with free variables $r_1 > r_2$, this is interpreted as:

every instance of r_1 is superior to every instance of r_2. Because, as we will see, the superiority relation is only used when it compares a literal and its negation, in general only a small portion of these superiority statements are significant.

2.2 Proof Theory

In the following $\sim p$ denotes the *complement* of p, that is, $\sim p$ is $\neg p$ if p is an atom, and $\sim p$ is q if p is $\neg q$.

Provability is based on the concept of a *derivation* (or *proof*) in $D = (F, R, >)$. A derivation is a finite sequence $P = (P(1), \ldots P(n))$ of tagged literals. A *tagged literal* consists of a sign ($+$ denotes provability, $-$ finite failure to prove), a tag and a literal.

The tags are not part of the object language; intuitively, they indicate the "strength" of the conclusions they are attached to, and correspond to different classes of derivations. Initially we will consider two tags: Δ denotes definite provability based on monotonic proofs, and *df* defeasible provability based on non-monotonic proofs.

The interpretation of the proof tags is as follows:

—$+\Delta q$: there is a definite derivation of q (i.e., a derivation using only strict rules and facts);

—$-\Delta q$: it is not possible to obtain a definite derivation of q (and this can be proved);

—$+df\, q$: there is a defeasible (non-monotonic) derivation of q;

—$-df\, q$: it is not possible to obtain a defeasible (non-monotonic) derivation of q (and this can be proved).

It is well known that there are different and sometimes incompatible intuitions behind what counts as a non-monotonic derivation. We will present a modular proof theory for defeasible logic and we will show how to modify some parameters to accommodate the above intuitions.

For a sequence of tagged literals to be a proof, it must satisfy certain conditions ($P(1..i)$ denotes the initial part of the sequence P of length i):

$+\Delta$) If $P(i+1) = +\Delta q$ then either
.1) $q \in F$; or
.2) $\exists r \in R_s[q]\ \forall a \in A(r), +\Delta a \in P[1..i]$.

That means, to prove $+\Delta q$ we need to establish a proof for q using facts and strict rules only. This is a deduction in the classical sense – no proofs for the negation of q need to be considered. Thus it is a *monotonic proof*.

$-\Delta$) If $P(i+1) = -\Delta q$ then
.1) $q \notin F$, and
.2) $\forall r \in R_s[q]\ \exists a \in A(r), -\Delta a \in P[1..i]$.

To prove $-\Delta q$, i.e. that q is not definitely provable, q must not be a fact. In addition, we need to establish that every strict rule with head q is *known to be* inapplicable. Thus for every such rule r there must be at least one antecedent a for which we have established that a is not definitely provable ($-\Delta a$).

The conditions to establish a *non-monotonic proof* have a structure similar to arguments, where we have three phases.

(1) In the first phase we have to put forward an argument for the conclusion we want to prove;
(2) in the second phase (the attack phase in argumentation terms) we have to consider all possible arguments against the thesis (counter-arguments);
(3) in the third and last phase we rebut the counter-arguments. To rebut an argument we have two options:
   (a) we can show that the argument is not founded, i.e., some of the premises do not hold, or
   (b) we can defeat the argument by providing a stronger counterargument.

In defeasible logic this is formalised as follows:

$+df$) If $P(i+1) = +df\ q$ then either
  .1) $+\Delta q \in P[1..i]$; or
  .2) The following three conditions all hold.
    .1) $-\Delta{\sim}q \in P[1..i]$, and
    .2) there is an applicable strict or defeasible rule for q, and
    .3) every rule for ${\sim}q$ is either
      .1) unsupported or
      .2) defeated.

Condition (1) says that every monotonic proof is also a non-monotonic proof. Condition (2.1) ensures that we cannot derive a conclusion non-monotonically while its negation is monotonically provable. For condition (2.2), a rule is applicable if it is either a strict or defeasible rule whose head is the conclusion we want to prove and all the literals in the antecedent are provable (in the rest of the paper we will discuss various degrees of provability, each of them corresponding to a different intuition). Clause (2.3) considers the possible counter-arguments, i.e., rules for the negation of the conclusion[1]. For a literal q to be derivable, each such counter-argument must be either unsupported or defeated. A rule is unsupported if it is not possible to give a (valid) justification for at least one of the premises of the rule. The degree of provability of the conclusion we want to obtain determines the meaning of valid justification for a premise. This could vary from a derivation for the premise to a simple chain of rules leading to it. Finally a rule is defeated if there is an applicable rule stronger than it.

The definition of the proof theory of Defeasible Logic is completed by the condition $-df$.

$-df$) If $P(i+1) = -df\ q$ then
  .1) $-\Delta q \in P[1..i]$, and
  .2) either

[1]Note that defeaters cannot be used to establish a positive conclusion (2.2). On the other hand, they are considered as possible counter-arguments in (2.3), so they may prevent a conclusion from being drawn. This treatment captures the intuitive interpretation of defeaters given in Subsection 2.1

.1) $+\Delta{\sim}q \in P[1..i]$, or
.2) every strict or defeasible rule for q is discarded or
.3) there is a rule for ${\sim}q$ such that
  .2) the rule is supported and
  .3) the rule is not defeated.

To show that it is not possible to have a defeasible derivation of a literal, first of all we have to ensure that it is not possible to have a definite derivation of it, condition (1). In addition of the following situation should be the case: There is a definite derivation of the complement (2.1); alternatively we have to look for the possible reasons to justify the conclusion. In Defeasible Logic a reason for a conclusion is just a strict or defeasible rule for it. Thus, to show we cannot derive the conclusion we have to show that for each rule, (2.2) we cannot fire it –i.e., the rule is discarded, meaning that we can show that at least one of the premises of the rule cannot be defeasibly provable– or that there is an at least equally strong argument for the complement of the conclusion we want to prove (2.3). This happens if there is a rule for the complement for which we can give a (valid) justification for every premises of it, i.e., the rule is supported, (2.3.2) and the rule is not defeated (2.3.3).

Notice that the condition $-df$ is nothing more than a strong negation of the condition $+df$ [Antoniou et al. 2000]. The strong negation of a formula is closely related to the function that simplifies a formula by moving all negations to an innermost position in the resulting formula and replace the positive tags with the respective negative tags and viceversa.

In the next section we briefly discuss some intuitions of non-monotonic reasoning and, based on the above structures, we present formal definitions of the conditions on proof corresponding to the various intuitions. The Defeasible Logic variants we are going to discuss in the next section –each capturing a different intuition– are instance of the abstract Defeasible logic presented here. The various instances are obtained by varying the concrete definitions of *applicable*, *discarded*, *supported*, *unsupported* and *defeated*. Since these notions are used in the concrete instances of $+df$ and $-df$, applicable and discarded are the strong negation of each other and so are supported and unsupported.

For a defeasible theory D, any tag d and any literal q we write $D \vdash +dq$ iff $+dq$ appears in a derivation in D and write $+d(D)$ to denote $\{q \mid D \vdash +dq\}$, and similarly for $-d$.

3. VARIANTS OF DEFEASIBLE LOGIC

Here we show how to instantiate the abstract proof theory of Section 2 with some concrete variants of defeasible logic.

3.1 Ambiguity Blocking vs Ambiguity Propagation

Intuitively, a literal is *ambiguous* if there is a (monotonic) chain of reasoning that supports a conclusion that p is true, another that supports that $\neg p$ is true, and the superiority relation does not resolve this conflict.

Example 1. Let us suppose that a piece of evidence A suggests that the defendant in a legal case is not responsible while a second piece of evidence B indicates that he/she is responsible; moreover the sources are equally reliable. According

to the underlying legal system a defendant is presumed innocent (i.e., not guilty) unless responsibility has been proved (without any reasonable doubt).

The above scenario is encoded in the following defeasible theory:

$$r_1 : \mathit{evidenceA} \Rightarrow \neg \mathit{responsible}, \qquad r_3 : \mathit{responsible} \Rightarrow \mathit{guilty},$$
$$r_2 : \mathit{evidenceB} \Rightarrow \mathit{responsible}, \qquad r_4 : \Rightarrow \neg \mathit{guilty}.$$

Given both *evidenceA* and *evidenceB*, the literal *responsible* is ambiguous. There are two applicable rules (r_1 and r_2) with the same strength, each supporting the negation of the other. As a consequence r_3 is not applicable, and so there is no applicable rule supporting the *guilty* verdict. This behaviour is called *ambiguity blocking*, since the ambiguity of *guilty* has been blocked by the failure to prove *responsible*. In contrast, *ambiguity propagation* describes a behaviour where ambiguity of a literal is propagated to dependent literals. If we propagate ambiguity then the literals *guilty* and $\neg$*guilty* are ambiguous; thus an undisputed conclusion cannot be drawn. On the other hand, if we assume an ambiguity blocking stance, the literal $\neg$*guilty* is not ambiguous and a verdict can be reached.

A preference for ambiguity blocking or ambiguity propagating behaviour is one of the properties of non-monotonic inheritance nets over which intuitions can clash [Touretzky et al. 1987]. Stein [1992] argues that ambiguity blocking results in an unnatural pattern of conclusions in extensions of the above example; e.g., if we extend it with the rule $r_5 : \neg \mathit{guilty} \Rightarrow \mathit{compensation}$, saying that in case of a not guilty verdict the defendant is entitled to compensation, then, despite the existence of equally strong pieces of evidence, she is entitled to compensation, while this is not the case in the ambiguity propagating case. See also Horty's [2002] discussion of ambiguity propagation.

The example and the discussion above show that both ambiguity blocking and ambiguity propagation can be used in the same application domain. In legal reasoning, typically, civil law takes an ambiguity blocking stance, while criminal law more often opts for an ambiguity propagation point of view. In addition there are situations where both types of reasoning are required at once. For example, in legal proceedings, often, the different parties involved have different burden of proof [Prakken and Sartor 2007], and these types call for different ways to lead to the conclusions. Typical, in criminal litigation, a plaintiff, to win a case, has to prove it without any benefit of doubt, while a defendant is only required to produce an exception to the accusation. This means that the claim of the plaintiff must be proved using ambiguity propagation reasoning, while the claim of the defendant by ambiguity propagation [Eriksson Lundström et al. 2007]. In addition the conditions to justify plaintiff claims are more stringent than those for the defendant.

It has been argued that ambiguity propagation results in fewer conclusions being drawn, in Section 5 we prove that this is indeed the case for the logics we discuss.

3.1.1 *Ambiguity Blocking Defeasible Logic.* We now instantiate the abstract definition of the proof theory given in Section 2 to an ambiguity blocking defeasible logic by defining what it means for a rule to be *applicable*, *discarded*, *supported*, *unsupported* and *defeated*. We will use the tag ∂ for ambiguity blocking defeasible

proofs (it instantiates df).

—A rule $r \in R[q]$, is *applicable* at step $i+1$ in a derivation P iff $\forall a \in A(r), +\partial a \in P[1..i]$; all the literals in the antecedent are defeasibly provable. A rule is *supported* iff it is *applicable*.

—A rule $r \in R[q]$ is *discarded* at step $i+1$ in a derivation P iff $\exists a \in A(r), -\partial a \in P[1..i]$; we have shown that it is not possible to prove at least one of the elements of the antecedent of the rule. A rule is *unsupported* iff the rule is *discarded*.

—A rule $r \in R[q]$ is *defeated* at step $i+1$ of a derivation P iff $\exists s \in R_{sd}[\sim q]$, such that $s > r$ and s is applicable; there is a stronger applicable rule for the complement of the head of the rule.

Notice that the resulting logic is the defeasible logic of [Antoniou et al. 2001], where a thorough analysis of the proof theory of the logic is presented.

Example 1. [(continued)] Given *evidenceA* and *evidenceB* as facts, both rules r_1 and r_2 are applicable, thus we can prove both $-\partial$*responsible* and $-\partial\neg$*responsible*. As a consequence rule r_3 is discarded while rule r_4 is vacuously applicable. Therefore we can prove $+\partial\neg$*guilty*.

3.1.2 *Ambiguity Propagating Defeasible Logic.* We can achieve ambiguity propagation behaviour by modifying the definition of the key notions of *supported* and *unsupported* rules: we make it more difficult to cast aside a competing rule, thus making it easier to block a conclusion. In Example 1, ambiguity was blocked by using the non-provability of *responsible* to discard rule r_3. We can achieve ambiguity propagating behaviour in this example by not rejecting r_3 though its antecedent is not provable. To do so we introduce a weak kind of proof, called *support*.

$+\int$) If $P(i+1) = +\int q$ then either
  .1) $+\Delta q \in P[1..i]$; or
  .2) there is a supported strict or defeasible rule r for q and
    for every rule s for $\sim q$ either
    .1) s is discarded or
    .2) s is not stronger than r.

$-\int$) If $P(i+1) = -\int q$ then
  .1) $-\Delta q \in P[1..i]$, and either
  .2) for every rule for q either
    .1) the rule is unsupported or
    .2) is defeated by an applicable rule.

Support for a literal p consists of a chain of reasoning that would lead us to conclude p in the absence of defeasibly derivable attack on some reasoning step.

Example 1. [(continued)] All literals (*responsible*, $\neg$*responsible*, $\neg$*guilty* and $\neg$*guilty*) are supported. But if $r_1 > r_2$ is added, then neither *responsible* nor *guilty* are supported.

We will use the tag δ for ambiguity propagating defeasible proofs (it instantiates df).

—A rule $r \in R[q]$ is *applicable* at step $i+1$ in a derivation P iff $\forall a \in A(r), +\delta a \in P[1..i]$, i.e., every literal in the antecedent is defeasibly provable.

—A rule is *discarded* iff $\exists a \in A(r), -\delta a \in p[1..i]$, i.e., at least one of the premises of the rule is not defeasibly provable.

—A rule $r \in R[q]$ is *supported* at step $i+1$ in a derivation P iff $\forall a \in A(r), +\int a \in P[1..i]$, i.e., there is a chain of support for every literal in the antecedent of the rule.

—A rule $r \in R[q]$ is *unsupported* at step $i+1$ in a derivation P iff $\exists a \in A(r), -\int a \in P[1..i]$, i.e., at least one of the premises of the rules is not supported.

With these definitions, we obtain the ambiguity propagating defeasible logic of [Antoniou et al. 2000].

An important difference between the ambiguity propagating and the ambiguity blocking defeasible logics is that the notion of applicable and supported coincides in the blocking variant (and so the notions of discarded and unsupported, being strong negation of the previous notions), while in the propagating variants these notions are distinct.

Example 1. [(continued)] All literals mentioned in the example (both positive and negated) are supported. As before, we conclude $-\partial$*responsible* and $-\partial\neg$*responsible*, since r_1 and r_2 overrule each other. However, in case of ambiguity propagation we conclude $-\delta\neg$*guilty*, since rule r_3 is only discarded but not unsupported. Thus a positive proof of $+\delta\neg$*guilty* is not possible. So overall we have both $-\delta\neg$*guilty* and $-\delta$*guilty*.

3.2 Team Defeat

The defeasible logics we have considered so far incorporate the idea of *team defeat* [Horty 2001]. That is, an attack on a rule with head p by a rule with head $\sim p$ may be defeated by a *different* rule with head p (see inference rule $+df$). Even though the idea of team defeat is natural, it is worth noting that it is not adopted by many related systems and concrete systems of argumentation. On the other hand, the notion of accrual of arguments [Prakken 2005] is gaining more prominence, and team defeat is a form of accrual (albeit one that can only strengthen the arguments in the team).

Example 2. Here we wish to give an example that illustrates the notion of teams.

$monotreme(platypus)$,	$laysEggs(platypus)$,
$hasFur(platypus)$,	$hasBill(platypus)$,
$r_1: \ monotreme(X) \Rightarrow mammal(X)$,	$r_3: \ laysEggs(X) \Rightarrow \neg mammal(X)$,
$r_2: \ hasFur(X) \Rightarrow mammal(X)$,	$r_4: \ hasBill(X) \Rightarrow \neg mammal(X)$,
$r_1 > r_3$,	$r_2 > r_4$.

Intuitively we conclude that a *platypus* is a *mammal* because for every reason against this conclusion (r_3 and r_4) there is a stronger reason for $mammal(platypus)$ (r_1 and r_2 respectively). It is easy to see that $+df \ mammal(platypus)$ is indeed provable in both the ambiguity blocking and the ambiguity propagating Defea-

sible Logic: there is a rule in support of $mammal(platypus)$, and every rule for $\neg mammal(platypus)$ is overridden by a rule for $mammal(platypus)$.

The authors believe that team defeat is a natural reasoning principle that should be respected. However, it is easy to define defeasible logics without team defeat in our framework. The only modification required is to change conditions $+df.2.3.2$ into "is defeated by r", and $-df.2.3.2$ to "is not defeated by r". The modified inference conditions are given in full in section A.2. The proof conditions are used in the formal theorem and its proof.

The change requires that to prove a conclusion we must have an applicable rule which is stronger than all applicable/non discarded rules for the negation of the conclusion we want to prove.

According to this new condition, in the above example, we no longer derive $+\partial mammal(platypus)$, since the applicable rule r_1 is not stronger than r_4 and similarly r_2 is not stronger than r_3.

Tags for inference rules without team defeat are expressed with a superscript *. Thus we will use ∂^*, δ^* and $\int^*$ to refer to the counterparts without team defeat of ∂, δ and $\int$.

4. COMMON PROPERTIES

In this section we discuss some properties common to the several logics described in the previous section and defined in Appendix A. In the following we refer to a number of tags, corresponding to the three levels of proof introduced:

(1) *Support*; we use the tag $\int$ to denote support with team defeat and $\int^*$ for that without team defeat.

(2) *Defeasible provability*; we use the tags ∂ and ∂^* for defeasible provability with and without team defeat for the ambiguity blocking variants, and we reserve the tags δ and δ^* for the same notions but in the ambiguity propagating variants.

(3) *Definite provability* is signaled by the tag Δ.

Our purpose here is not to provide detailed proofs of the properties for each logic – that would be repetitious and take away from the focus of this paper. Instead we will indicate how the properties can be established.

The commonality of these properties stems largely from each logic being an instance of the defeasible logic framework of [Antoniou et al. 2000], based on the meta-program formulation in [Maher and Governatori 1999]. We begin by outlining this framework, and then address the properties.

4.1 The Defeasible Logic Framework

The framework is based around a meta-program defined in logic programming with unit clauses used to represent the facts, rules, and superiority relation of a defeasible theory. The defeasible theory is expressed through predicates `fact`, `strict`, `strict_or_defeasible`, `rule` and `sup` describing, respectively, the facts, strict rules, strict and defeasible rules, all rules, and the superiority relation. We give below the meta-program for the logic consisting of Δ and ∂ tags. These two tags are represented by the predicates `definitely` and `defeasibly`. We write $M(D)$

for the combination of the meta-program and the representation of the defeasible theory D.

Comparison of the clauses of this meta-program with clauses in the proof rules for Δ and ∂ given in Section 2 and Appendix A.2 should make clear that the meta-program and the proof rules have the same structure. The only complication is that each universal quantifier in the proof rules is expressed with negations and (an implicit) existential quantifier. We leave it to the reader to express the proof rules for the other tags as meta-programs.

$c1$ `definitely`(X):-
  `fact`(X).

$c2$ `definitely`(X):-
  `strict`$(R, X, [Y_1, \ldots, Y_n])$,
  `definitely`(Y_1),...,`definitely`(Y_n).

$c3$ `defeasibly`(X):-
  `definitely`(X).

$c4$ `defeasibly`(X):-
  `not definitely`$(\sim X)$,
  `strict_or_defeasible`$(R, X, [Y_1, \ldots, Y_n])$,
  `defeasibly`(Y_1),...,`defeasibly`(Y_n),
  `not overruled`(R, X).

$c5$ `overruled`(R, X):-
  `rule`$(S, \sim X, [U_1, \ldots, U_n])$,
  `defeasibly`(U_1),...,`defeasibly`(U_n),
  `not defeated`$(S, \sim X)$.

$c6$ `defeated`$(S, \sim X)$:-
  `sup`(T, S),
  `strict_or_defeasible`$(T, X, [V_1, \ldots, V_n])$,
  `defeasibly`(V_1),...,`defeasibly`(V_n).

The meaning of the meta-program is determined by Kunen's [1987] semantics of logic programs. `defeasibly`(p) is a consequence of $M(D)$ iff $D \vdash +\partial q$; $\neg$`defeasibly`(p) is a consequence of $M(D)$ iff $D \vdash -\partial q$ [Maher and Governatori 1999; Antoniou et al. 2000].

4.2 Coherence

Coherence of a defeasible logic refers to the property that tagged literals obtained by applying complementary tags to the same literal cannot both be inferred.

Definition 1. *A logic is* coherent *if, for every defeasible theory D in the logic, every tag d, and every literal q, we do not have both $D \vdash +dq$ and $D \vdash -dq$.*

It is this property that supports the intuition that $+d$ represents a form of provability while $-d$ represents finite failure to prove.

All the logics addressed in this paper are instances of the framework. Consequently, by Theorem 2 of [Antoniou et al. 2000], they are coherent. We can also draw this conclusion directly from the framework using the fact that, under Kunen's semantics, a logic program cannot entail both p and $\neg p$.

PROPOSITION 2. *Consider a defeasible theory D. If $d \in \{\Delta, \delta, \partial, \int, \delta^*, \partial^*, \int^*\}$ then $+d(D) \cap -d(D) = \emptyset$.*

4.3 Consistency

In common with most monotonic logics, the monotonic part of a defeasible theory can produce inconsistency. For example, the theory involving facts p and $\neg p$ can infer both $+\Delta p$ and $+\Delta\neg p$. Since the defeasible proof rules often directly adopt conclusions of the monotonic part, defeasible conclusions may also be inconsistent. The underlying cause of inconsistency in such cases is the monotonic part of the logic, so consistency for defeasible tags refers to the capability of defeasible proof rules to avoid introducing any further inconsistencies.

DEFINITION 3. *A tag d in a logic is* consistent *if, for every defeasible theory D in the logic and every proposition q, we do not have $D \vdash +dq$ and $D \vdash +d\neg q$ unless $D \vdash +\Delta q$ and $D \vdash +\Delta\neg q$.*

Consistency was established for ∂ in [Billington 1993]. By our containment theorem (Theorem 7), it follows that δ and δ^* are consistent. ∂^* cannot be proved consistent in this way, but a direct proof is relatively straightforward. It uses the proof rule $+\partial^*$ to explore how $+\partial^* q$ and $+\partial^*\neg q$ could both hold. Using the coherence of the logic and the acyclicity of the superiority relation, all possibilities are excluded except for $+\Delta q$ and $+\Delta\neg q$.

PROPOSITION 4. *For $d \in \{\delta, \partial, \delta^*, \partial^*\}$, d is consistent.*

However, neither $\int$ nor $\int^*$ are consistent. Consider the rules

$$\begin{aligned} r: &\quad \Rightarrow p, \\ r': &\quad \Rightarrow \neg p. \end{aligned}$$

Then we can conclude $+\int p$ and $+\int \neg p$, and the same for $\int^*$.

4.4 Decisiveness

Decisiveness is a form of inverse of coherence: where coherence requires that at most one of $+dq$ and $-dq$ is inferred, decisiveness requires that at least one is inferred. Decisiveness insists that the proof theory determines the proof status of every tagged literal.

DEFINITION 5. *A logic is* decisive *if, for every defeasible theory D in the logic, every tag d, and every literal q, either $D \vdash +dq$ or $D \vdash -dq$.*

By design, defeasible logics are not decisive. Inferring neither provability nor non-provability allows the logics to express situations where the status of a tagged literal is "unknown". Non-decisiveness also allows the proof system to react sensibly to theories such as

$$\begin{aligned} r_1: &\quad \Rightarrow p, \\ r_2: &\ q \Rightarrow \neg p, \\ r_3: &\quad \Rightarrow q, \\ r_4: &\ p \Rightarrow \neg q, \end{aligned}$$

where (under ∂ and ∂^*) $+dp$ can be concluded iff $-dq$ can be concluded, and $+dq$ can be concluded iff $-dp$ can be concluded. The symmetry of p and q in this theory suggests that a decisive logic must make an arbitrary choice between $+dp$ and $+dq$.

4.5 Computational Complexity

While the meta-program is different for each logic in this paper, all logics use Kunen's semantics for logic program. It follows that the first order form of each logic is computable, as observed in [Antoniou et al. 2000]. In contrast, most first-order non-monotonic logics are not computable.

For the propositional form of the logics, we can unfold the clauses of the meta-program by the atoms representing the defeasible theory facts, rules and superiority relation. This semantics-preserving operation results in a variable-free, essentially propositional logic program. If performed naively this operation can have quadratic cost but, by employing a single structure for each unfolding of $c6$ by a rule T that is shared by all S such that $T > S$, the representation becomes linear. Inference of all literals that are consequences of a propositional logic program under Kunen's semantics can be performed in linear time. Consequently

PROPOSITION 6. *Consider a propositional defeasible theory D. If $d \in \{\Delta, \delta, \partial, \int, \delta^*, \partial^*, \int^*\}$ then $+d(D)$ and $-d(D)$ can be computed in time linear in the size of D.*

An alternative approach to proving this result is to adapt the more detailed proof of the linear complexity of ∂ [Maher 2001].

5. INCLUSION THEOREM

In this section we wish to establish relationships among the provability concepts introduced in this paper. The main theorem of this paper shows that there exists a chain of increasing expressive power among several of the logics.

The following theorem formalises the inclusion property.

THEOREM 7 (INCLUSION THEOREM).

(a) $+\Delta(D) \subseteq +\delta^*(D) \subseteq +\delta(D) \subseteq +\partial(D) \subseteq +\int(D) \subseteq +\int^*(D)$.
(b) $-\int^*(D) \subseteq -\int(D) \subseteq -\partial(D) \subseteq -\delta(D) \subseteq -\delta^*(D) \subseteq -\Delta(D)$.
(c) For each inclusion relationship in (a) and (b) there exists a theory D such that the inclusion is strict.

An important consequence of this theorem is that defeasible logic exhibits different grades of inference, representing different degrees of cautiousness in drawing conclusions. It is interesting that this is achieved without using numerical weights such as probabilities or uncertainty factors.

Other formalisms also have various grades of inference. In particular, there is the well known hierarchy of logic programming semantics given by the sequence of Kunen, Fitting, Well-Founded and Stable semantics, related to the interpretation of negation as failure. There is a similar hierarchy in semantics of argumentation systems (see, for example, [Dung et al. 2007]). The logic programming hierarchy is also reflected in the class of defeasible logics, since the application of these semantics to the meta-program of Section 4 gives us a hierarchy of inferences. This hierarchy is orthogonal to the hierarchy identified in Theorem 7: one reflects the hierarchy of

semantics applied to the meta-program while the other is derived from variations in the form of the meta-program. Hence, defeasible logic provides great flexibility in expressing inferences with various degrees of caution.

The inclusions *(a)* and *(b)* in Theorem 7 look expected. For example, the relation $+\delta^*(D) \subseteq +\delta(D)$ appears trivial since the absence of team defeat makes the inference rule weaker. But notice that there is a potential source of complication: when the logic fails to prove a literal p and instead shows its non-provability, then that result may be used by the logic to prove another literal q that could not be proven if p were provable. In fact, while δ^* is weaker than δ, as Theorem 7 states; ∂^* is *not* weaker than defeasible provability with team defeat (∂). Consider the following example:

Example 3.

$$r_1 : \Rightarrow p, \qquad r_3 : \Rightarrow \neg p, \qquad r_5 : p \Rightarrow \neg q,$$
$$r_2 : \Rightarrow p, \qquad r_4 : \Rightarrow \neg p, \qquad r_6 : \Rightarrow q,$$
$$r_1 > r_3,\ r_2 > r_4.$$

Here rules r_1, r_2, r_3, and r_4 are all applicable. Rules r_1 and r_2 form a team that defeats r_3 and r_4, thus we have $+\partial p$, and then both r_5 and r_6 are applicable and there is no way of solving the conflict, thus we conclude $-\partial q$. On the other hand, no applicable rule for p is stronger than all applicable rules for $\neg p$, thus we have $-\partial^* p$. This makes rule r_5 discarded, while rule r_6 is still applicable, therefore we can conclude $+\partial^* q$.

Clearly, then, an apparent strengthening of an inference rule may have unexpected consequences; the class of logics is also non-monotonic. This makes the chain of inclusions of Theorem 7 even more striking.

It also suggests caution in adapting existing formalisms to support accrual of arguments [Prakken 2005]. If such a primitive notion of accrual as team defeat can have such varying affects (consider δ, ∂ and $\int$) then a more sophisticated notion can also be expected to behave differently when added to different inference mechanisms.

There are some technical consequences of the Inclusion Theorem that are significant. As we have seen in Section 3.1.1, in legal reasoning ambiguity blocking and ambiguity propagation can be used in the same application. Moreover in the application modelling litigation, it is important that the claims for the plaintiff (i.e., those justified using ambiguity blocking) are a subset of those for the defendant (i.e., those justified using ambiguity propagation), since all claims by the plaintiff must be available to the defendant to use in a counter-argument. Thus the establishment of $+\delta(D) \subseteq +\partial(D)$ supports the use of defeasible logic to model litigation. It also provides some further formal support for the folklore that ambiguity propagating behaviour results in fewer conclusions being drawn, compared with ambiguity blocking behaviour.

At a purely technical level, the theorem allowed us to establish the consistency of the ambiguity propagating inference, with or without team defeat, in Section 4.3.

Although our results here apply only to defeasible logic, they are indicative of similar results for other formalisms. In [Governatori et al. 2004] we have shown that

Dung's grounded argumentation semantics [Dung 1995] characterises the ambiguity propagating variants of defeasible logic, while the ambiguity blocking variants require more expressiveness than is available in Dung's abstract model of argumentation. This suggests that nonmonotonic formalisms that have been characterized in terms of Dung's grounded argumentation semantics are ambiguity propagating. More generally, it suggests the many formalisms characterized by argumentation semantics might admit both ambiguity propagating and blocking variants having relationships similar to Theorem 7. Investigation of these issues is left for future research.

The notions of ambiguity blocking and ambiguity propagation apply to situations where one has no criterion (e.g., preference relation, weighting, ...) to resolve a conflict between complementary conclusions. On the other hand the notion of team defeat is strongly related to the ability to express preferences on rules, in the case of defeasible logic via the superiority relation. There are numerous works that involve rules and preferences, including LPwNF [Dimopoulos and Kakas 1995], courteous logic programs [Grosof 1997], defaults with priorities [Horty 2007], priority rules [Wang et al. 1996], CR-Prolog [Balduccini and Gelfod 2003], preference logic programming [Govindarajan et al. 1995], weak constraints in DLV [Buccafurri et al. 1997]. The first two of these are closely related to defeasible logic [Antoniou et al. 2000], and our results should be directly relevant to them. The third employs a notion of uniform defeat, but also briefly explores a form of team defeat. The others apply preferences globally rather than locally to individual inferences, and it appears team defeat is not relevant in such cases.

Most formulations of argumentation employ an attack relation between individual arguments; for such formulations, team defeat cannot be expressed[2]. However, recent formulations of argumentation [Bochman 2003; Nielsen and Parsons 2006] permit attack between sets of arguments, as, of course, did the original argumentation characterization of defeasible logic [Governatori and Maher 2000]. For these formulations, team defeat is expressible and our results are relevant.

In summary, the Inclusion Theorem establishes some useful structure among the many variants of defeasible logic. It is also suggestive of similar structures in other non-monotonic formalisms which remain to be investigated.

6. CONCLUSION

This paper introduced several variants of defeasible logic, based on an abstract presentation of the proof theory. These variants capture different intuitions about reasoning issues: ambiguity blocking versus propagation, and team defeat. The main contribution of this paper is the presentation and proof of the Inclusion Theorem which shows that different concepts of provability form a chain of increasing strength.

REFERENCES

Antoniou, G. and Bikakis, A. 2007. DR-Prolog: A system for defeasible reasoning with rules and ontologies on the semantic web. *IEEE Transaction in Knowledge and Data Engineering 19,* 2,

[2]The notion of acceptability (or support) in abstract argumentation involves a relation between a single argument and a set of arguments, but it still does not admit two teams.

233–245.

Antoniou, G., Billington, D., Governatori, G., and Maher, M. J. 2000. A flexible framework for defeasible logics. In *17th National Conference on Artificial Intelligence and 12th Conference on on Innovative Applications of Artificial Intelligence (AAAI/IAAI)*. AAAI Press / The MIT Press, 405–410.

Antoniou, G., Billington, D., Governatori, G., and Maher, M. J. 2001. Representation results for defeasible logic. *ACM Transactions on Computational Logic 2,* 2, 255–287.

Antoniou, G., Billington, D., Governatori, G., and Maher, M. J. 2006. Embedding defeasible logic into logic programming. *Theory and Practice of Logic Programming 6,* 6, 703–735.

Antoniou, G., Billington, D., Governatori, G., Maher, M. J., and Rock, A. 2000. A family of defeasible reasoning logics and its implementation. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, W. Horn, Ed. IOS Press, 459–463.

Antoniou, G., Maher, M. J., and Billington, D. 2000. Defeasible logic versus logic programming without negation as failure. *Journal of Logic Programming 42,* 1, 47–57.

Balduccini, M. and Gelfod, M. 2003. Logic programs with consistency-restoring rules. In *International Symposium on Logical Formalization of Commonsense Reasoning*, P. Doherty, J. McCarthy, and M.-A. Williams, Eds. AAAI 2003 Spring Symposium Series. AAAI, 9–18.

Bassiliades, N., Antoniou, G., and Vlahavas, I. P. 2006. A defeasible logic reasoner for the semantic web. *International Journal on Semantic Web and Information Systems 2,* 1, 1–41.

Billington, D. 1993. Defeasible logic is stable. *Journal of Logic and Computation 3,* 4, 379–400.

Bochman, A. 2003. Collective argumentation and disjunctive logic programming. *Journal of Logic and Computation 13,* 3, 405–428.

Buccafurri, F., Leone, N., and Rullo, P. 1997. Strong and weak constraints in disjunctive datalog. In *4th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'97)*, J. Dix, U. Furbach, and A. Nerode, Eds. Lecture Notes in Computer Science, vol. 1265. Springer, 2–17.

Dimopoulos, Y. and Kakas, A. C. 1995. Logic programming without negation as failure. In *International Symposium on Logic Programming*. MIT Press, 369–383.

Dumas, M., Governatori, G., ter Hofstede, A. H. M., and Oaks, P. 2002. A formal approach to negotiating agents development. *Electronic Commerce Research and Applications 1,* 2, 193–207.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence 77,* 2, 321–358.

Dung, P. M., Mancarella, P., and Toni, F. 2007. Computing ideal sceptical argumentation. *Artificial Intelligence 171,* 10-15, 642–674.

Eriksson Lundström, J., Governatori, G., Thakur, S., and Padmanabhan, V. 2007. An asymmetric protocol for argumentation games in defeasible logic. In *10th Pacific Rim International Conference on Multi-Agents (PRIMA 2007)*, A. K. Ghose, G. Governatori, and R. Sadananda, Eds. Lecture Notes in Computer Science, vol. 5044. Springer, 219–231.

Governatori, G. 2005. Representing business contracts in RuleML. *International Journal of Cooperative Information Systems 14,* 2-3, 181–216.

Governatori, G. and Maher, M. J. 2000. An argumentation-theoretic characterization of defeasible logic. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, W. Horn, Ed. IOS Press, 469–473.

Governatori, G., Maher, M. J., Billington, D., and Antoniou, G. 2004. Argumentation semantics for defeasible logics. *Journal of Logic and Computation 14,* 5, 675–702.

Governatori, G. and Rotolo, A. 2008. BIO logical agents: Norms, beliefs, intentions in defeasible logic. *Autonomous Agents and Multi-Agent Systems 17,* 1, 36–69.

Governatori, G., Rotolo, A., and Padmanabhan, V. 2006. The cost of social agents. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, H. Nakashima, M. P. Wellman, G. Weiss, and P. Stone, Eds. 513–520.

Governatori, G., Rotolo, A., and Sartor, G. 2005. Temporalised normative positions in defeasible logic. In *10th International Conference on Artificial Intelligence and Law (ICAIL 2005)*. ACM, 25–34.

Govindarajan, K., Jayaraman, B., and Mantha, S. 1995. Preference logic programming. In *12th International Conference on Logic Programming*. MIT Press, 731–745.

Grosof, B. N. 1997. Prioritized conflict handling for logic programs. In *Proceedings of the 1997 International Logic Programming Symposium*. MIT Press, 197–211.

Grosof, B. N., Labrou, Y., and Chan, H. Y. 1999. A declarative approach to business rules in contracts: courteous logic programs in xml. In *ACM Conference on Electronic Commerce*. 68–77.

Horty, J. F. 2001. Argument construction and reinstatement in logics for defeasible reasoning. *Artificial Intelligence and Law 9,* 1, 1–28.

Horty, J. F. 2002. Skepticism and floating conclusions. *Artificial Intelligence 135,* 1-2, 55–72.

Horty, J. F. 2007. Defaults with priorities. *Journal of Philosophical Logic 36*, 367–413.

Kraus, S., Lehmann, D. J., and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence 44,* 1-2, 167–207.

Kunen, K. 1987. Negation in logic programming. *Journal of Logic Programming 4,* 4, 289–308.

Maher, M. J. 2001. Propositional defeasible logic has linear complexity. *Theory and Practice of Logic Programming 1,* 6, 691–711.

Maher, M. J. 2002. A model-theoretic semantics for defeasible logic. In *Proceedings of Workshop on Paraconsistent Computational Logic*. 67–80.

Maher, M. J. and Governatori, G. 1999. A semantic decomposition of defeasible logics. In *16th National Conference on Artificial Intelligence and 11th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI)*. 299–305.

Makinson, D. 1994. General patterns in nonmonotonic reasoning. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. II*, D. Gabbay, Ed. Oxford University Press, 35–110.

McCarthy, J. 1980. Circumscription — a form of non-monotonic reasoning. *Artificial Intelligence 13,* 1-2, 27–39.

Moore, R. C. 1985. Semantical considerations on nonmonotonic logic. *Artificial Intelligence 25,* 1, 75–94.

Nielsen, S. H. and Parsons, S. 2006. A generalization of dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *3rd International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2006)*, N. Maudet, S. Parsons, and I. Rahwan, Eds. Lecture Notes in Computer Science, vol. 4766. Springer, 54–73.

Nute, D. 1994. Defeasible logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. III*, D. Gabbay, Ed. Oxford University Press, 353–395.

Prakken, H. 1997. *Logical Tools for Modelling Legal Argument: A Study of Defeasible Reasoning in Law*. Kluwer Academic Publishers.

Prakken, H. 2005. A study of accrual of arguments, with applications to evidential reasoning. In *10th International Conference on Artificial Intelligence and Law (ICAIL 2005)*. ACM, 85–94.

Prakken, H. and Sartor, G. 2007. Formalising arguments about the burden of persuasion. In *11th International Conference on Artificial Intelligence and Law (ICAIL 2007)*. 97–106.

Reeves, D. M., Wellman, M. P., and Grosof, B. N. 2002. Automated negotiation from declarative contract descriptions. *Computational Intelligence 18,* 4, 482–500.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence 13,* 1-2, 81–132.

Stein, L. A. 1992. Resolving ambiguity in nonmonotonic inheritance hierarchies. *Artificial Intelligence 55,* 2, 259–310.

Touretzky, D. S., Horty, J. F., and Thomason, R. H. 1987. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In *Proceedings of IJCAI*. 476–482.

Wang, X., You, J.-H., and Yuan, L.-Y. 1996. Nonmonotonic reasoning by monotonic inference with priority constraints. In *Non-Monotonic Extensions of Logic Programming (NMELP'96)*, J. Dix, L. M. Pereira, and T. C. Przymusinski, Eds. Lecture Notes in Computer Science, vol. 1216. Springer, 91–109.

A. PROOF OF THEOREM 7

A.1 Proof Approach and Outline

The proof of Theorem 7 is organised as follows. Section A.2 provides the expanded inference conditions, to be used as reference in the remainder of the Appendix. Section A.3 provides a number of technical preliminaries. Finally, Section A.4 provides the proof of Theorem 7, using a sequence of technical lemmas that correspond to specific inclusion relations.

The proof of Theorem 7 is *constructive*: an inclusion $+d_1(T) \subseteq +d_2(T)$ is shown by transforming a derivation for $+d_1 p$ to a derivation for $+d_2 p$; similar for an inclusion $-d_1(T) \subseteq -d_2(T)$.

Section A.4 contains a number of lemmas. Each of these lemmas corresponds to specific inclusion relations of Theorem 7(a) and Theorem 7(b) and define appropriate *derivation transformations*. Except for statements that follow directly from the definitions, the correctness of each derivation transformation is proven by induction on the length of a derivation.

The proof of the lemmas is of varying difficulty. Lemmas 11 and 12 are the most difficult to prove, as they address the inclusions incorporating team defeat. As a consequence, one reasoning chain may be countered by a different reasoning chain. That this cannot go on forever is shown by a regression argument, which shows that each time a different reasoning chain comes into play, an increasing chain of rule comparisons $\ldots > r_4 > r_3 > r_2 > r_1$ is built. Due to the finiteness of defeasible theories and the acyclicity of the relation $>$, this chain cannot become infinite, and the desired conclusion is drawn (see proof of Lemma 11 for technical details). In contrast, Lemmas 13 and 14 do not require this reasoning, because an attack on a rule cannot be countered by a different rule.

A.2 Expanded Proof Conditions

We first present the fully expanded definitions of the proof conditions presented in the main part of the paper for the convenience of the reader. Notice that in some cases, some sub conditions have been included to facilitate their identification in the course of proofs of the following lemmas.

$+\partial$) If $P(i+1) = +\partial q$ then either
- .1) $+\Delta q \in P[1..i]$; or
- .2) The following three conditions all hold.
  - .1) $\exists r \in R_{sd}[q]\ \forall a \in A(r), +\partial a \in P[1..i]$, and
  - .2) $-\Delta{\sim}q \in P[1..i]$, and
  - .3) $\forall s \in R[{\sim}q]$ either
    - .1) $\exists a \in A(s), -\partial a \in P[1..i]$; or
    - .2) $\exists t \in R_{sd}[q]$ such that
      - .1) $\forall a \in A(t), +\partial a \in P[1..i]$, and
      - .2) $t > s$.

$-\partial$) If $P(i+1) = -\partial q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) either
.1) $\forall r \in R_{sd}[q]\ \exists a \in A(r), -\partial a \in P[1..i]$; or
.2) $+\Delta{\sim}q \in P[1..i]$; or
.3) $\exists s \in R[{\sim}q]$ such that
.1) $\forall a \in A(s), +\partial a \in P[1..i]$, and
.2) $\forall t \in R_{sd}[q]$ either
.1) $\exists a \in A(t), -\partial a \in P[1..i]$; or
.2) not$(t > s)$.

$+\delta$) If $P(i+1) = +\delta q$ then either
.1) $+\Delta q \in P[1..i]$; or
.2) The following three conditions all hold.
.1) $\exists r \in R_{sd}[q]\ \forall a \in A(r), +\delta a \in P[1..i]$, and
.2) $-\Delta{\sim}q \in P[1..i]$, and
.3) $\forall s \in R[{\sim}q]$ either
.1) $\exists a \in A(s), -\int a \in P[1..i]$; or
.2) $\exists t \in R_{sd}[q]$ such that
.1) $\forall a \in A(t), +\delta a \in P[1..i]$, and
.2) $t > s$.

$-\delta$) If $P(i+1) = -\delta q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) either
.1) $\forall r \in R_{sd}[q]\ \exists a \in A(r), -\delta a \in P[1..i]$; or
.2) $+\Delta{\sim}q \in P[1..i]$; or
.3) $\exists s \in R[{\sim}q]$ such that
.1) $\forall a \in A(s), +\int a \in P[1..i]$, and
.2) $\forall t \in R_{sd}[q]$ either
.1) $\exists a \in A(t), -\delta a \in P[1..i]$; or
.2) not$(t > s)$.

$+\int$) If $P(i+1) = +\int q$ then either
.1) $+\Delta q \in P[1..i]$; or
.2) $\exists r \in R_{sd}[q]$ such that
.1) $\forall a \in A(r), +\int a \in P[1..i]$, and
.2) $\forall s \in R[{\sim}q]$ either
.1) $\exists a \in A(s), -\delta a \in P[1..i]$; or
.2) not$(s > r)$.

$-\int$) If $P(i+1) = -\int q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) $\forall r \in R_{sd}[q]$ either
.1) $\exists a \in A(r), -\int a \in P[1..i]$; or
.2) $\exists s \in R[\sim q]$ such that
.1) $\forall a \in A(s), +\delta a \in P[1..i]$, and
.2) $s > r$.

$+\partial^*$) If $P(i+1) = +\partial^* q$ then either
.1) $+\Delta q \in P[1..i]$; or
.2) $\exists r \in R_{sd}[q]$ such that
.1) $\forall a \in A(r), +\partial^* a \in P[1..i]$, and
.2) $-\Delta \sim q \in P[1..i]$, and
.3) $\forall s \in R[\sim q]$ either
.1) $\exists a \in A(s), -\partial^* a \in P[1..i]$; or
.2) $r > s$.

$-\partial^*$) If $P(i+1) = -\partial^* q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) $\forall r \in R_{sd}[q]$ either
.1) $\exists a \in A(r), -\partial^* a \in P[1..i]$; or
.2) $+\Delta \sim q \in P[1..i]$; or
.3) $\exists s \in R[\sim q]$ such that
.1) $\forall a \in A(s), +\partial^* a \in P[1..i]$, and
.2) not$(r > s)$.

$+\delta^*$) If $P(i+1) = +\delta^* q$ then either
.1) $+\Delta q \in P[1..i]$; or
.2) $\exists r \in R_{sd}[q]$ such that
.1) $\forall a \in A(r), +\delta^* a \in P[1..i]$, and
.2) $-\Delta \sim q \in P[1..i]$, and
.3) $\forall s \in R[\sim q]$ either
.1) $\exists a \in A(s), -\int^* a \in P[1..i]$; or
.2) $r > s$.

$-\delta^*$) If $P(i+1) = -\delta^* q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) $\forall r \in R_{sd}[q]$ either
.1) $\exists a \in A(r), -\delta^* a \in P[1..i]$; or
.2) $+\Delta \sim q \in P[1..i]$; or
.3) $\exists s \in R[\sim q]$ such that
.1) $\forall a \in A(s), +\int^* a \in P[1..i]$, and
.2) not$(r > s)$.

$+\int^*$) If $P(i+1) = +\int^* q$ then either
.1) $+\Delta q \in P[1..i]$; or
.2) $\exists r \in R_{sd}[q]$ such that
.1) $\forall a \in A(r), +\int^* a \in P[1..i]$, and
.2) $\forall s \in R[\sim q]$ either
.1) $\exists a \in A(s), -\delta^* a \in P[1..i]$; or
.2) not$(s > r)$.

$-\int^*$) If $P(i+1) = -\int^* q$ then
.1) $-\Delta q \in P[1..i]$, and
.2) $\forall r \in R_{sd}[q]$ either
.1) $\exists a \in A(r), -\int^* a \in P[1..i]$; or
.2) $\exists s \in R[\sim q]$ such that
.1) $\forall a \in A(s), +\delta^* a \in P[1..i]$, and
.2) $s > r$.

A.3 Technical Preliminaries

Since derivations are sequences, we shall need several functions which deal with sequences. Let () denote the empty sequence. Any non-empty sequence, P, can be denoted by $(h|T)$ where h is the head (or first element) of P and T is the tail (or rest) of P.

& denotes *concatenation*. We say P' is a *prefix* of P iff there exists a sequence P'' such that $P = P'\&P''$. Sometimes we write PQ instead of $P\&Q$.

Postpend, denoted by $+$, is a function which takes a sequence and an element and attaches the element onto the right end of the sequence.

DeleteAll is a function which takes an element and a sequence and deletes every occurrence of that element from the sequence.

Interleave, denoted by $Intlv$, is a function which takes two sequences and produces the set of all interleavings of these two sequences. We say P'' is an interleaving of P and P' iff $P'' \in Intlv(P, P')$.

$Intlv((), P) = \{P\}$. $Intlv((h|T), ()) = \{(h|T)\}$.

$Intlv((h_1|T_1), (h_2|T_2)) = \{(h_1|X) : X \in Intlv(T_1, (h_2|T_2))\} \cup \{(h_2|X) : X \in Intlv((h_1|T_1), T_2)\}$.

Concatenation is an extreme form of interleaving where the sequences do not mix. At the other extreme is alternation, where the sequences are mixed as much as possible.

Alternate, denoted by Alt, is a function which takes two sequences and alternates the elements of each. $Alt((), P) = P$. $Alt((h|T), ()) = (h|T)$. $Alt((h_1|T_1), (h_2|T_2)) = (h_1, h_2)\&Alt(T_1, T_2)$. *Reduce* is a function which takes a sequence and removes all second and subsequent occurrences of elements.

$Reduce(P)$ is called the *reduced form* of P. If $P = (P(1), ..., P(n))$ is a sequence and $j \in [1..n]$ we define $P - j$ to be the sequence obtained from P by just deleting the element in position j.

Suppose d_1 and d_2 are tags. If E is a derivation or a tagged literal in a derivation then we define $E(d_1 := d_2)$ to be the result of just replacing all occurrences of d_1 in E by d_2. Furthermore we assume that the application of the substitution $d_1 := d_2$ has higher precedence than concatenation and postpending.

Let P be a derivation and let S be any set of tags. Define PS to be the subsequence of P formed by just deleting all the elements of P which do not contain a tag in S. So every tagged literal in PS contains a tag in S, and every tagged literal in P which contains a tag in S is in PS.

LEMMA 8.
(1) Any prefix of any derivation is also a derivation.
(2) Any interleaving of any two derivations is also a derivation. In particular, the insertion of one derivation into another is also a derivation; the concatenation of two derivations is also a derivation; and the alternation of two derivations is also a derivation.
(3) If $P = (P(1), ..., P(n))$ is a derivation and $P(i) = P(j)$ and $i < j$ then $P - j$ is a derivation.
(4) The reduced form of any derivation is also a derivation.
(5) If $P\&P'$ is a derivation and $P\&P''$ is a derivation then $P\&P'\&P''$ is a derivation.
(6) If P is a derivation and $e \in P$ then $P + e$ is a derivation.
(7) If P is a derivation then each of $P\{+\Delta\}$, $P\{-\Delta\}$, $P\{+\Delta, -\Delta, +\delta, -\int\}$, $P\{+\Delta, -\Delta, +\partial, -\partial\}$, $P\{+\Delta, -\Delta, +\int, -\delta\}$, $P\{+\Delta, -\Delta, +\partial^*, -\partial^*\}$, $P\{+\Delta, -\Delta, +\delta^*, -\int^*\}$, and $P\{+\Delta, -\Delta, +\int^*, -\delta^*\}$ is a derivation.

PROOF. Parts 1, 2, 3, 6, and 7 follow directly from the definition of a derivation. Part 4 follows from repeated application of part 3. Part 5 follows from repeated application of part 3 to $P\&P'\&P\&P''$, which is a derivation by part 2. □

A.4 Proof of the Inclusion Theorem

The following two lemmas follow directly from the definition of the derivation conditions.

LEMMA 9. *Let P be a derivation in a defeasible theory T, q a literal, and $d \in \{\delta, \partial, \int, \delta^*, \partial^*, \int^*\}$. If $+\Delta q \in P$ then $P\&(+dq)$ is a derivation in T. So $+\Delta(T) \subseteq +d(T)$.*

LEMMA 10. *Let P be a derivation in a defeasible theory T, q a literal, and $d \in \{\delta, \partial, \int, \delta^*, \partial^*, \int^*\}$. If $-dq \in P$ then $-\Delta q \in P$. So $-d(T) \subseteq -\Delta(T)$.*

LEMMA 11. *If P is a derivation in a defeasible theory $T = (F, R, >)$ then $P\&Alt(P(+\partial := +\int), P(-\partial := -\delta))$ is a derivation in T. So $+\partial(T) \subseteq +\int(T)$ and $-\partial(T) \subseteq -\delta(T)$.*

PROOF. The proof is by induction on the length of P. The lemma is trivially true for the empty derivation P.

Suppose the lemma holds for all derivations of length $\leq n$. Let P be a derivation of length $n + 1$. Then $P[1..n]\&Alt(P[1..n](+\partial := +\int), P[1..n](-\partial := -\delta))$ is a derivation. Since P is a derivation, by Lemma 8(5), $Q := P\&Alt(P[1..n](+\partial := +\int), P[1..n](-\partial := -\delta))$ is a derivation. Then $P\&Alt(P(+\partial := +\int), P(-\partial := -\delta)) = Q + P(n + 1)(+\partial := +\int) + P(n + 1)(-\partial := -\delta)$.

If $P(n+1) = dq$ where d is a tag and $d \notin \{+\partial, -\partial\}$ then $P(n+1)(+\partial := +\int) = P(n+1) = P(n+1)(-\partial := -\delta)$ and so $Q+P(n+1)(+\partial := +\int)+P(n+1)(-\partial := -\delta)$ is a derivation by Lemma 8(6).

Case 1: $P(n+1) = +\partial q$.

Then $P(n+1)(+\partial := +\int) = +\int q$, and $P(n+1)(-\partial := -\delta) = P(n+1)$, so $Q + P(n+1)(+\partial := +\int) + P(n+1)(-\partial := -\delta) = Q\&((+\int q) + P(n+1)$. If $+\partial.1$ holds then, by $+\int.1$, $Q\&(+\int q)$ is a derivation.

If $+\partial.1$ fails then $+\partial.2$ holds. Then there is $r \in R_{sd}[q]$ such that for all a in $A(r)$, $+\partial a \in P[1..n]$. Therefore $+\int a \in P[1..n](+\partial := +\int)$ and hence $+\int a \in Q$.

Hence $+\int.2.1(P[1..i] = Q)$ holds. If $+\int.2.2(P[1..i] = Q)$ holds then $Q\&(+\int q)$ is a derivation. If $+\int.2.2(P[1..i] = Q)$ fails then there exists $s \in R[\sim q]$ such that $s > r$ and for all a in $A(s)$, $-\delta a \notin Q$. By $+\partial.2.3$, either (i) there is a in $A(s)$ such that $-\partial a \in P[1..n]$; or (ii) there exists r_1 in $R_{sd}[q]$ such that $r_1 > s$ and for all $a \in A(r_1)$, $+\partial a \in P[1..n]$. If (i) holds then $-\delta a \in P[1..n](-\partial := -\delta)$ which contradicts $-\delta a \notin Q$. Hence (ii) holds. Therefore, for all $a \in A(r_1)$, $+\int a \in P[1..n](+\partial := +\int)$ and hence $+\int a \in Q$. So $r_1 > s > r$, and for all $a \in A(r_1)$, $+\int a \in Q$.

But $+\int.2.1(r = r_1, P[1..i] = Q)$ now holds. If $\int.2.2(r = r_1, P[1..i] = Q)$ holds then $Q\&(+\int q)$ is a derivation. If $\int.2.2(r = r_1, P[1..i] = Q)$ fails then there exists $s_1 \in R[\sim q]$ such that $s_1 > r_1$ and for all a in $A(s_1)$, $-\delta a \notin Q$. By $+\partial.2.3(r = r_1)$, either (i) there exists a in $A(s_1)$ such that $-\delta a \in P[1..n]$; or (ii) there exists $r_2 \in R_{sd}[q]$ such that $r_2 > s_1$ and for all $a \in A(r_2)$, $+\partial a \in P[1..n]$. If (i) then $-\delta a \in P[1..n](-\partial := -\delta)$ which contradicts $-\delta a \notin Q$. Hence (i) holds. Therefore for all a in $A(r_2)$, $+\int a \in P[1..n](+\partial := +\int)$ and hence $+\int a \in Q$. So $r_2 > s_1 r_1 > s > r$, and for all a in $A(r_2)$, $+\int a \in Q$.

A suitable number of applications of the same reasoning as in the preceding paragraph shows that either $Q\&(+\int q)$ is a derivation, or there is an infinite chain $\ldots > r_3 > s_2 > r_2 > s_1 > r_1 > s > r$, where r and r_i are in $R_{sd}[q]$ and s and s_i are in $R[\sim q]$. But such a chain cannot exist because $R_{sd}[q]$ is finite and $>$ is acyclic. Thus $Q\&(+\int q)$ is a derivation.

Thus in all subcases $Q\&(+\int q)$ is a derivation, and so by Lemma 8(6), $Q\&(+\int q)+P(n+1)$ is a derivation.

Case 2: $P(n+1) = -\partial q$.

Then $P(n+1)(+\partial := +\int) = P(n+1)$, and $P(n+1)(-\partial := -\delta)$, so $Q + P(n+1)(+\partial := +\int) + P(n+1)(-\partial := -\delta) = Q + P(n+1)\&(-\delta q)$. By Lemma 8(6), $Q + P(n+1)$ is a derivation, which we shall denote by Q'.

By $-\partial.1$, $-\delta.1(P[1.i] = Q')$ holds. If $-\partial.2.1$ holds then $-\delta.2.1(P[1..i] = Q')$ holds. If $-\partial.2.2$ holds then $-\delta.2.2(P[1..i] = Q')$ holds.

If $-\partial.2.3$ holds then there exists s in $R[\sim q]$ such that for all a in $A(s)$, $+\partial a \in P[1..n]$. Then $+\int a \in P[1..n](+\partial := +\int)$ and so $+\int a \in Q'$. Hence $-\delta.2.3.1(P[1..i] = Q')$ holds. Moreover, for all $t \in R_{sd}[q]$, either $t > s$ does not hold, or there exists a in $A(t)$ such that $-\partial a \in P[1..n]$. Therefore $-\delta.2.3.2(P[1..i] = Q')$ holds and hence $-\delta.2.3(P[1..i] = Q')$ holds.

So $Q'\&(-\delta q)$ is a derivation.

Thus the lemma is proved by induction. □

LEMMA 12. *If P is a derivation in a defeasible theory T then $P\&Alt(P(-\int := -\partial), P(+\delta := +\partial))$ is a derivation in T. So $+\delta(T) \subseteq +\partial(T)$ and $-\int(T) \subseteq -\partial(T)$.*

PROOF. As for Lemma 11. □

LEMMA 13. *If P is a derivation in a defeasible theory $T = (F, R, >)$ then*

$P(+\delta^*, -\int^* := +\delta, -\int)$ is a derivation in T. So $+\delta^*(T) \subseteq +\delta(T)$ and $-\int^*(T) \subseteq -\int(T)$.

PROOF. The proof is by induction on the length of a derivation P. The claim trivially holds for length zero. So let P have length $n+1$. Then $Q = P[1..n](+\delta^*, -\int^* := +\delta, -\int)$ is a derivation. Then $P(+\delta^*, -\int^* := +\delta, -\int) = Q + P(n+1)(+\delta^*, -\int^* := +\delta, -\int)$.

If $P(n+1) = dq$, where d is a tag other than $+\delta^*$ and $-\int^*$, then $P(n+1)(+\delta^*, -\int^* := +\delta, -\int) = P(n+1)$, thus $Q + P(n+1)(+\delta^*, -\int^* := +\delta, -\int)$ is a derivation by Lemma 8(7).

Case 1: $P(n+1) = +\delta^* q$.

Then $P(n+1)(+\delta^*, -\int^* := +\delta, -\int) = +\delta q$, so $Q + P(n+1)(+\delta^*, -\int^* := +\delta, -\int) = Q\&(+\delta q)$. If $+\delta^*.1(P[1..i] = P[1..n])$ holds then $\delta.1(P[1..i] = Q)$ holds, and so $Q\&(+\delta q)$ is a derivation. If $+\delta^*.1(P[1..i] = P[1..n])$ fails then $+\delta^*.2(P[1..i] = P[1..n])$ holds.

Since $+\delta^*.2(P[1..i] = P[1..n])$ holds, there exists $r \in R_{sd}[q]$ such that for all $a \in A(r)$, $+\delta^* a \in P[1..n]$. Therefore $+\delta a \in P[1..n](+\delta^*, -\int^* := +\delta, -\int)$, and hence $+\delta.2.1(P[1..i] = Q)$ holds.

Since $+\delta^*.2.2(P[1..i] = P[1..n])$ holds, $+\delta.2.2(P[1..i] = Q)$ is true.

Take any $s \in R[\sim q]$. By $+\delta^*.2.3(P[1..i] = P[1..n])$, either (i) there exists $a \in A(s)$ such that $-\int^* a \in P[1..n]$; or (ii) $r > s$. If (i) then $-\int a \in P[1..n](+\delta^*, -\int^* := +\delta, -\int)$, and so $+\delta.2.3.1(P[1..i] = Q)$ holds. If (ii) then $+\delta.2.3.2(P[1..i] = Q, t = r)$. Since s was arbitrary, $+\delta.2.3(P[1..i])$ holds.

Thus $Q\&(+\delta q)$ is a derivation.

Case 2: $P(n+1) = -\int^* q$.

Then $P(n+1)(+\delta^*, -\int^* := +\delta, -\int) = -\int q$, so $Q + P(n+1)(+\delta^*, -\int^* := +\delta, -\int) = Q\&(-\int q)$. By $-\int^*.1(P[1..i] = P[1..n])$, $-\int.1(P[1..i] = Q)$ holds. Take any $r \in R_{sd}[q]$.

If $-\int^*.2.1(P[1..i] = P[1..n])$ then there is $a \in A(r)$ such that $-\int^* a \in P[1..n]$. So $-\int a \in Q$ and hence $-\int.2.1(P[1..i] = Q)$ holds.

If $-\int^*.2.2$ holds then there exists $s \in R[\sim q]$ such that for all $a \in A(s)$, $+\delta^* a \in P[1..n]$ and $s > r$.

So $+\delta a \in Q$, hence $-\int.2.2(P[1..i] = Q)$ holds.

Since r was arbitrary, $-\int.2(P[1..i] = Q)$ holds, and so $Q\&(-\int q)$ is a derivation.

Overall, the lemma is proved by induction. □

LEMMA 14. *If P is a derivation in a defeasible theory T then $P(-\delta, +\int := -\delta^*, +\int^*)$ is a derivation in T. So $+\int(T) \subseteq +\int^*(T)$ and $-\delta(T) \subseteq -\delta^*(T)$.*

PROOF. As for Lemma 13. □

PROOF OF THEOREM 7.

For part (a):

$+\Delta(T) \subseteq$	Lemma 9
$+\delta^*(T) \subseteq$	Lemma 13
$+\delta(T) \subseteq$	Lemma 12
$+\partial(T) \subseteq$	Lemma 11
$+\int(T) \subseteq$	Lemma 14
$+\int^*(T)$.	

For part (b):

$-\int^*(T) \subseteq$ Lemma 13
$-\int(T) \subseteq$ Lemma 12
$-\partial(T) \subseteq$ Lemma 11
$-\delta(T) \subseteq$ Lemma 14
$-\delta^*(T) \subseteq$ Lemma 10
$-\Delta^*(T)$.

For part (c): The strict inclusions for (a) are shown as follows. For $+\Delta(T) \subset +\delta^*(T)$ we consider the theory consisting of a single defeasible rule $\Rightarrow a$. Clearly, $-\Delta a$ as there is no strict rule in the theory, and so $a \notin +\Delta(T)$, but $+\partial^* a$. For $+\delta^*(T) \subset +\delta(T)$ one can use the theory of Example 2. For $+\delta(T) \subset +\partial(T)$ this is shown by the theory of Example 1 where we have $+\partial\neg guilty$ but $-\delta\neg guilty$. For $+\partial(T) \subset +\int(T)$ we can use again Example 1 where we have $-\partial responsible$ and $+\int responsible$. Finally for $+\int(T) \subset +\int^*(T)$ we can use Example 3 where we have $-\int q$ but $+\int^* q$.

The strict inclusions for (b) are shown as follows. For $-\int^*(T) \subset -\int(T)$ consider the theory of Example 3, extended by adding $r_5 > r_6$ to the superiority relation. Then $+\delta p$ and not $+\delta^* p$, so $q \in -\int(T)$ and $q \notin -\int^*(T)$. For $-\int(T) \subset -\partial(T)$ consider the theory T with just the rules $\Rightarrow q$ and $\Rightarrow \neg q$ (the superiority relation is empty). Then $q \in -\partial(T)$ but $q \notin -\int(T)$. For $-\partial(T) \subset -\delta(T)$ consider the theory of Example 1. We can prove $+\int responsible$ but not $+\partial responsible$. Hence $\neg guilty \in -\delta(T)$ and $\neg guilty \notin \partial(T)$. For $-\delta(T) \subset -\delta^*(T)$ consider the theory of Example 3. We can prove $-\delta^* p$ but not $-\delta p$. Finally, for $-\delta^*(T) \subset -\Delta(T)$ consider the theory consisting of the single rule $\Rightarrow q$. Then $-\Delta a$ is provable, but not $-\delta^* q$. □

Received March 2008; revised January 2009; accepted June 2009