Entailment Semantics for Rules with Priorities

David Billington
Institute for Integrated and Intelligent Systems
School of Information and Communication Technology
Nathan campus, Griffith University
Brisbane, Queensland 4111, Australia.
d.billington@griffith.edu.au

Abstract

We define a new general rule-based non-monotonic framework which allows an external acyclic priority relation between rules to be interpreted in several ways. Several entailment semantics are defined via a constructive digraph, with one being given a declarative fixed-point characterisation as well. Each of these semantics satisfies Principle 1 of [Brewka and Eiter 1999]. The framework encompasses Default Logic [Reiter 1980], ground Answer Set Programming (ASP) [Baral 2003], and Defeasible Logic [Nute 1994]. Default Logic is provided with a new semantics which is ambiguity blocking, rather than the usual ambiguity propagating semantics. Also Reiter-extensions are given a new fixed-point characterisation; and Lukaszewicz's [1990] m-extensions are given a much simpler construction and fixed-point characterisation.

1 Introduction

A characteristic of an intelligent system is its ability to reason with limited information. Non-monotonic reasoning systems represent and reason with limited information where the limitation is not quantified. Some of the most natural and successful non-monotonic reasoning systems are based on the idea of a rule. Three such systems are Default Logic [Reiter 1980], Answer Set Programming (ASP) [Baral 2003], and Defeasible Logic [Nute 1994].

After considering these systems we shall develop a general rule-based non-monotonic framework, called a plausible theory, which encompasses these three systems. Plausible theories allow an external acyclic priority relation between rules to be interpreted in several ways. Several entailment semantics for plausible theories will be defined via a constructive digraph, with one being given a declarative fixed-point characterisation as well. (An entailment semantics defines the set of all sentences that follow from, or are entailed by, a plausible theory.) Each of these semantics satisfies Principle 1 of [Brewka and Eiter 1999].

Default Logic is provided with a new semantics which is ambiguity blocking, rather than the usual ambiguity propagating semantics. So for ambiguity blockers, there is now a Default Logic which gets the "right" answer for Example 4.1. Also Reiter-extensions are given a new fixed-point characterisation; and Lukaszewicz's [1990] m-extensions are given a much simpler construction and fixed-point characterisation.

The unifying framework facilitates comparison of the systems encompassed. The constructive semantics facilitates understanding, and hand calculation of small examples.

In this paper we shall concentrate on the relationship between plausible theories and the Default Logics of Reiter [1980] and Lukaszewicz [1990]. The relationship with other Default Logics, ASP, Defeasible Logic, and Argumentation systems [Dung 1995] is a subject for other papers.

Now let us look at the above systems to see the variety of rules that have been considered.

Reiter's Default Logic considers a rule to be a default. A (closed) *default*, d, can be written as $d = (p(d) : J(d) / c(d))$, where $p(d)$, the pre-requisite of d, and $c(d)$, the consequent of d, are closed formulas, and $J(d)$, the set of justifications of d, is a non-empty finite set of closed formulas. Intuitively the meaning of d is that if $p(d)$ is accepted and each element of $J(d)$ is consistent with what is currently accepted, then accept $c(d)$.

An ASP rule has the form

$$l_0 \text{ or } ... \text{ or } l_k \leftarrow l_{k+1}, ..., l_m, \text{not}l_{m+1}, ..., \text{not}l_n$$

where the l_i are literals. Intuitively the meaning of such a rule is that if $l_{k+1}, ..., l_m$, are accepted and $l_{m+1}, ..., l_n$ can be safely assumed to be false then accept at least one of $l_0, ..., l_k$. A program is a finite set of ASP rules.

Nute's Defeasible Logic has three kinds of rules. The strict rule, r, has the form $A(r) \rightarrow c(r)$; the defeasible rule, r, has the form $A(r) \Rightarrow c(r)$; and the defeater rule, r, has the form $A(r) \leadsto c(r)$; where $A(r)$, the set of antecedents of r, is a finite set of ground literals and $c(r)$, the consequent of r, is a ground literal. Strict rules are like classical material implication. Intuitively defeasible rules mean that if $A(r)$ is true then usually $c(r)$ is true. Defeater rules have the intuitive meaning that if $A(r)$ is true then it is too risky to suppose $\neg c(r)$ is true.

All of the above rules have a set of antecedents and a consequent. However defaults and some ASP rules have an additional consistency checking part which we shall call a set of guards.

It is very useful to be able to specify a priority between rules. Defeasible Logic has always had this ability. More recently preferences between defaults and between ASP rules have been considered. See [Delgrande et al 2004] for a survey of the area and a classification of priorities. According to this classification our priority relation is acyclic, between rules, and external.

The rest of the paper has the following organisation. The next section formally defines a system of rules, called a plausible theory, which encompasses Default Logic, ground ASP programs, and Defeasible Logic. Section 3 discusses semantics, proposes several constructive ways of defining what a plausible theory entails, and then shows that a principle concerning priorities is satisfied. Each construction forms a directed graph called an applicative digraph. An example illustrating ambiguity blocking and propagating is given in Section 4. Section 5 concerns the relationship between applicative digraphs and Default Logic. In particular Reiter's extensions and Lukaszewicz's [1990] m-extensions are characterised. Section 6 considers a fixed-point characterisation of the leaves of an applicative digraph. The conclusions form Section 7.

2 Plausible Theories

We set the scene by defining some notation. The empty set is denoted by {}, and $\mathbb{N}$ denotes the set of non-negative integers. Let *Snt* be the set of sentences (closed formulas) of a countable first-order language. Then *Snt* is countable. If $S \subseteq Snt$ then define the set, $Cn(S)$, of *semantic consequences* of *S* by $Cn(S) = \{s \in Snt : S \models s\}$.

Definition 2.1 (rule). A *rule*, *r*, is a 4-tuple
$(A(r), G(r), arrow(r), c(r))$, where
$A(r)$, the set of *antecedents* of *r*, is a finite set of sentences,
$G(r)$, the set of *guards* of *r*, is a finite set of sentences,
$arrow(r) \in \{\Rightarrow, \rightsquigarrow\}$, and
$c(r)$, the *consequent* of *r*, is a sentence.
Rules which use the $\Rightarrow$ arrow are called *defeasible* rules and are written as $A(r){:}G(r) \Rightarrow c(r)$.
Rules which use the $\rightsquigarrow$ arrow are called *warning* rules and are written as $A(r){:}G(r) \rightsquigarrow c(r)$.

Intuitively the defeasible rule, *r*, means that if each element of $A(r)$ is accepted and each element of $G(r)$ is consistent with what is currently accepted, then $c(r)$ should be accepted. In other words, *r* provides sufficient evidence so that one should behave as if $c(r)$ is true, even though it may turn out that $c(r)$ is false.

So the default $(p(d) : J(d) / c(d))$ can be regarded as the defeasible rule $\{p(d)\}{:}J(d) \Rightarrow c(d)$. The ground ASP rule

$$l_0 \text{ or } ... \text{ or } l_k \leftarrow l_{k+1}, ..., l_m, \text{not}l_{m+1}, ..., \text{not}l_n$$

can be regarded as the defeasible rule

$$\{l_{k+1}, ..., l_m\}{:}\{\neg l_{m+1}, ..., \neg l_n\} \Rightarrow l_0 \vee ... \vee l_k.$$

And Nute's defeasible rule $A(r) \Rightarrow c(r)$ can be regarded as our defeasible rule $A(r){:}\{c(r)\} \Rightarrow c(r)$.

Intuitively the warning rule, *r*, warns that if each element of $A(r)$ is accepted and each element of $G(r)$ is consistent with what is currently accepted, then accepting $\neg c(r)$ is too risky. That is, *r* provides enough evidence for $c(r)$ to prevent acceptance of $\neg c(r)$, but not sufficient evidence to accept $c(r)$. Warning rules are only used to stop other rules being applied. They never contribute their consequent to what is currently accepted. The defeater rule $A(r) \rightsquigarrow c(r)$ can be regarded as the warning rule $A(r){:}\{c(r)\} \rightsquigarrow c(r)$.

Recall that a default, *d*, is *normal* iff $J(d) = \{c(d)\}$. So we have the following definition.

Definition 2.2 (normal rule). A rule *r* is *normal* iff $G(r) = \{c(r)\}$. If *r* is normal then we often abbreviate $A(r){:}G(r) \Rightarrow c(r)$ by $A(r) \Rightarrow c(r)$, and $A(r){:}G(r) \rightsquigarrow c(r)$ by $A(r) \rightsquigarrow c(r)$.

Let *R* be any set of rules. We denote the set of defeasible rules in *R* by R_d, and the set of warning rules in *R* by R_w. The set of consequents of *R* is denoted by $c(R)$; that is $c(R) = \{c(r) : r \in R\}$. To aid readability we sometimes write $c[R]$ for $c(R)$. The set of all guards in rules in *R* is denoted by $G(R)$; that is $G(R) = \cup\{G(r) : r \in R\}$.

A binary relation, >, on any set *R* is *cyclic* iff there exists a sequence, $(r_1, r_2, ..., r_n)$ where $n \geq 1$, of elements of *R* such that $r_1 > r_2 > ... > r_n > r_1$. A relation is *acyclic* iff it is not cyclic. Acyclicity is about the weakest property that an ordering relation can have. Acyclicity implies irreflexivity and asymmetry (and hence antisymmetry); however we do not require transitivity.

Let > be an acyclic binary relation on a set of rules *R*. We read $r > r'$ as *r has a higher priority than r'*, or *r* is *superior* to *r'*. We say *r* is *maximal in R* iff for all *r'* in *R*, not$(r' > r)$.

Definition 2.3 (plausible theory). A *plausible theory* is a triple $\Pi = (Ax, R, >)$ such that PT1, PT2, and PT3 all hold.
(PT1) *Ax* is a satisfiable set of sentences.
(PT2) *R* is a set of rules.
(PT3) > is an acyclic binary relation on *R*.
$\Pi = (Ax, R, >)$ is a *normal* plausible theory iff every rule in *R* is normal.

The sentences in *Ax* are called *axioms* and they characterise the aspects of the situation which are certain. *R* is countable because *Snt* is countable.

3 Entailment Semantics

Let us start by considering what properties a semantics for a plausible theory might have. Predicate logic has two parts, a proof mechanism and a semantics. The semantics interprets the syntax used by the proof mechanism, and it also provides a way of assuring us that the proof mechanism cannot prove things we don't want (soundness).

In this paper we are concerned with the semantics of plausible theories, rather than any proof mechanisms.

Applicative digraphs (defined below) provide an interpretation of the rules and symbols used in a plausible theory. As the name "applicative" suggests, we interpret rules as things which can be applied. The way they are applied and the restrictions on their application give an interpretation to the syntactic components of the rules.

The way a semantics assures us that the proof mechanism is sound is to define a set, say *Sem*(*P*), of sentences that follow from the premisses P and then insist that every sentence that is proved must be in $Sem(P)$. In predicate logic $Sem(P)$ is $Cn(P)$. An *entailment semantics* for P is the set of all sentences which follow from, or are entailed by, P. For predicate logic this is generally agreed to be $Cn(P)$. An *entailment semantics* for a plausible theory Π is the set, say $Sem(\Pi)$, of all sentences which follow from, or are entailed by, Π. Since entailment is an intuitive notion, the definition of $Sem(\Pi)$ needs to be as close to our intuitions as possible.

There are different well-informed intuitions about what should follow from a plausible theory, for instance the ambiguity propagating intuition differs from the ambiguity blocking intuition (see example 4.1). To accommodate this we shall define a family of applicative digraphs.

Applicative digraphs enable $Sem(\Pi)$ to be constructed; so no guessing is needed.

3.1 Applicative Digraphs

An applicative digraph displays the order in which rules are applied. Each node is the set of rules applied so far, hence each node is a subset of R. Before applicative digraphs can be defined we need the following concepts. Suppose $\Pi = (Ax, R, >)$ is a plausible theory, $r \in R$, and $N \subseteq R$.

Definition 3.1 ($B(N)$, the *belief set* of N).
$B(N) = Cn(Ax \cup c(N))$.

$B(N)$ is the set of beliefs sanctioned by Ax and N. At each node $N \subseteq R$ there are 3 levels of belief. The set of strongest beliefs is $Cn(Ax)$. This is the set of known facts, indisputably true statements. It is the same at each node. $B(N_d)$ is the set of all beliefs which are strong enough to behave as if they are true. Even though we know that some may occasionally be false. The set of weakest beliefs at N is $B(N)$. This is the set of all beliefs which are strong enough to prevent belief in any statement which is inconsistent with $B(N)$.

Definition 3.2 ($Ap(r, N)$, r is *applicable* at N; $Ap(N)$).
$Ap(r, N)$ iff $A(r) \subseteq B(N_d)$.
$Ap(N)$ iff for all r in N, $Ap(r, N)$.

The fundamental idea is that if every antecedent of a rule is in $B(N_d)$ then its consequent can be added to $B(N)$. This is what applying r to $B(N)$ means. However we do not apply a rule which will conflict with our current set of weakest beliefs; so we need the idea of a set of rules being consistent.

Definition 3.3 ($Cs(N)$, N is *consistent*).
$Cs(N)$ iff $Ax \cup c(N)$ is satisfiable and
for all g in $G(N)$, $Ax \cup c(N) \cup \{g\}$ is satisfiable.

Definition 3.4 ($El(r, N)$, r is *eligible* at N).
$El(r, N)$ iff $Ap(r, N)$ and $Cs(N \cup \{r\})$.

$El(r, N)$ is a useful summary term.

According to Delgrande and Schaub [2000] priorities may be prescriptive (dictating the order in which rules are applied, which has a local perspective) or descriptive (describing the preferred results of applying rules, which requires a global perspective).

A local perspective on $>$ is given by $Pr(r, N)$, which means that maximal rules are applied before non-maximal rules, and that higher priority rules are applied before lower priority rules.

Definition 3.5 ($Pr(r, N)$, r has *priority* at N).
$Pr(r, N)$ iff either r is maximal in R;
or r is not maximal in R and for each s in $R-(N \cup \{r\})$,
if $El(s, N)$ then not$(s > r)$ and s is not maximal in R.

Before we can define a global perspective on $>$ we need the idea of a set, M, of rules being compatible; which roughly means that M could be a subset of a node.

Definition 3.6 ($Cp(M)$, M is *compatible*).
$Cp(M)$ iff there exists N such that
$M \subseteq N \subseteq R$ and $Ap(N)$ and $Cs(N)$.
$Cp(r_1, ..., r_n)$ means $Cp(\{r_1, ..., r_n\})$.

We can now define what it means for a set N of rules to be desirable. Roughly, if $r \notin N$ and r is applicable at N and r is superior to a rule in N then r is incompatible with a higher priority rule already in N.

Definition 3.7 ($Ds(N)$, N is *desirable*; $Ds(r, N)$).
$r > N$ iff there exists r' in N such that $r > r'$.
$Ds(N)$ iff for all r in $R-N$, if $Cp(r)$, $Ap(r, N)$, and $r > N$
then there exists r' in N such that $r' > r$ and not $Cp(r', r)$.
$Ds(r, N)$ iff $Ds(N \cup \{r\})$.

If two rules could both be applied at a node, but the application of either rule prevents the application of the other rule, then we may not want to apply either rule. That is we may not want to choose between conflicting rules, unless one is preferred over the other. Of course if it does not matter which of two rules is applied first then we can apply either. This is the notion of partner rules. Roughly, two rules are partners at node N means that if either rule could be applied to a potential descendant Q of N, then both rules could be applied to Q.

Definition 3.8 ($Pa(r, r', N)$, r and r' are *partners* at N).
$Pa(r, r', N)$ iff $El(r, N)$, $El(r', N)$, and for each Q such that
$N \subseteq Q \subseteq R$ and $Cs(Q)$ and $Ap(Q)$,
if $Cs(Q \cup \{r\})$ or $Cs(Q \cup \{r'\})$ then $Cs(Q \cup \{r, r'\})$.

There are many ways to decide if rule r is preferred over rule s. The simplest is if $r > s$. However a more general notion of "defeat by a partner" is more useful. Basically if r and s conflict but a partner of r has a higher priority than s then we prefer r over s. This leads to the idea of there being a winner at a node.

Definition 3.9 ($Wn(r, N)$, r is a *winner* at N).
$Wn(r, N)$ iff $El(r, N)$, $Pr(r, N)$, and for each s in $R-(N \cup \{r\})$,
if $El(s, N)$ and not $Cs(N \cup \{r, s\})$ then
there exists r' in $R-N$ such that
$El(r', N)$ and $Pa(r, r', N)$ and $r' > s$.

Depending on one's intuition one may use Pr, Wn, Ds, or none of them at each node. Hence the following definition.

Definition 3.10 ($0(r, N)$).
$0(r, N)$ is true.

If every rule that can be applied to $B(N_d)$ is in N then we say N is full. There are several notions of fullness.

Definition 3.11 ($Full(I, N)$, N is *I-full*).
Suppose $I \in \{0, Pr, Wn, Ds\}$. $Full(I, N)$ iff for each r in R, if $El(r, N)$ and $I(r, N)$ then $r \in N$.

Excluding infinite nodes, an applicative digraph is a tree in which some non-sibling nodes which are the same distance from the root maybe coalesced. We shall therefore use tree nomenclature to describe parts of these digraphs. The node M is called a *parent* of the node N, and N is called a *child* of M, iff there is an arc from M to N. A node which has no children is called a *leaf*. A path starting at the root is called a *branch*.

Every finite non-empty node has at least one parent, but infinite nodes have no parents and no children.

Definition 3.12 ($App(I, \Pi)$). For each I in $\{0, Pr, Wn, Ds\}$, the *I-applicative digraph*, $App(I, \Pi)$, of a plausible theory $\Pi = (Ax, R, >)$ is defined as follows.
The empty set, $\{\}$, is the *root* node of $App(I, \Pi)$.
For each finite node N of $App(I, \Pi)$ and each r in R,
$N \cup \{r\}$ is a child of N iff $r \notin N$, $El(r, N)$, and $I(r, N)$.
If $N \cup \{r\}$ is a child of N then the arc from N to $N \cup \{r\}$ is labelled with r.
Every finite node of $App(I, \Pi)$ and every arc in $App(I, \Pi)$ comes from the above construction.
Suppose $N_0 = \{\}$ and $(N_0, r_1, N_1, r_2, N_2, ...)$ is an infinite path in $App(I, \Pi)$ and $N = \cup\{N_i : i \in \mathbb{N}\}$. Then N is a node in $App(I, \Pi)$ iff $Full(I, N)$.

The purpose of an applicative digraph is to construct all its desirable leaves. It is the belief set of the defeasible rules in a desirable leaf that we want. There are now two choices. The credulous position is to regard each such belief set as entailed by Π. The sceptical position is to regard only the intersection of all such belief sets as entailed by Π.

Definition 3.13 ($Sem(I, \Pi)$).
For each I in $\{0, Pr, Wn, Ds\}$, the *I-semantics* of Π, $Sem(I, \Pi)$, is defined by
$Sem(I, \Pi) = \cap\{B(N_d) : N \text{ is a leaf of } App(I, \Pi) \text{ and } Ds(N)\}$.

3.2 Principles for Priorities on Rules

Brewka and Eiter [1999] present two principles which they said should be satisfied by any system of prioritised defeasible rules. Delgrande et al [2004] show that descriptive priorities do not satisfy the second principle, but they agree that the first principle is widely accepted. We shall re-cast the first principle so that it is simpler and more general.

Principle 1.
Let r_1 and r_2 be two rules and suppose N is a set of rules. If r_1 is preferred over r_2 then $N \cup \{r_2\}$ should not be preferred over $N \cup \{r_1\}$.

It can be shown that if $N \cup \{r_1\}$ is a node in an applicative digraph then not $Ds(N \cup \{r_2\})$. So the above definition of desirable leaves satisfies Principle 1. A reasonable alternative to $Ds(N)$ is $Pr(N)$, where

$Pr(N)$ iff for all r in N, $Pr(r, N-\{r\})$.

But if $N \cup \{r_1\}$ is a node in an applicative digraph then not $Pr(r_2, N)$. So Principle 1 holds if $Pr(N)$ replaces $Ds(N)$.

4 Example

In this example and elsewhere if the set of antecedents of a rule is a singleton set we shall often omit its set braces.

We say $K(N)$ is a *kernel* of $B(N)$ iff
(i) $K(N) \subseteq B(N)$, and
(ii) $Cn(K(N)) = B(N)$, and
(iii) if $S \subseteq K(N)$ and $S \neq K(N)$ then $Cn(S) \neq B(N)$.
When drawing applicative digraphs for given examples it is useful to write $K(N)$ underneath each node N.

In the following example, rather than drawing an applicative digraph with kernels, which takes a lot of space, we shall list its arcs using the following linear notation. The arc labelled r from node N with kernel $K(N)$ to node $N \cup \{r\}$ with kernel $K(N \cup \{r\})$ is denoted by

$[N, K(N)]$--r-->$[N \cup \{r\}, K(N \cup \{r\})]$.

Example 4.1 (Ambiguity Propagating or Blocking)

This example is designed to show the difference between the ambiguity propagating intuition and the ambiguity blocking intuition. Consider the following five statements, and determine whether e is likely or not. Each defeasible rule is followed by its symbolic form in brackets.
(1) a, b, and c are facts.
(2) If a then usually e. $[a \Rightarrow e]$
(3) If b then usually d. $[b \Rightarrow d]$
(4) If c then usually not d. $[c \Rightarrow \neg d]$
(5) If d then usually not e. $[d \Rightarrow \neg e]$

A plausible theory describing this situation is $\Pi = (Ax, R, >)$ where $Ax = \{a, b, c\}$, $R = \{a \Rightarrow e,\ b \Rightarrow d,\ c \Rightarrow \neg d,\ d \Rightarrow \neg e\}$, and $>$ is empty. For easy reference, let r_1 be $a \Rightarrow e$, r_2 be $b \Rightarrow d$, r_3 be $c \Rightarrow \neg d$, and r_4 be $d \Rightarrow \neg e$.

Intuitively, given Ax, r_2 is evidence for d and r_3 is equal evidence for $\neg d$. So d and $\neg d$ are ambiguous, and hence neither should follow from Π. r_1 is evidence for e. The only evidence against e is r_4, but r_4 is weakened by the ambiguity of d. If you think that it is too risky to allow e to follow from Π then e will be ambiguous and so the ambiguity of d has propagated along r_4 to e. On the other hand if you think that r_4 has been sufficiently weakened to allow e to follow from Π then e will not be ambiguous and so the ambiguity of d has been blocked from propagating along r_4. Both ambiguity blocking and ambiguity propagating intuitions are common, and so a semantics should allow for both.

A linear representation of $App(0, \Pi)$, with kernels, is:
$[\{\}, \{a,b,c\}]$--r_1-->$[\{r_1\}, \{a,b,c,e\}]$
  $[\{r_1\}, \{a,b,c,e\}]$--r_2-->$[\{r_1,r_2\}, \{a,b,c,d,e\}]$
  $[\{r_1\}, \{a,b,c,e\}]$--r_3-->$[\{r_1,r_3\}, \{a,b,c,\neg d,e\}]$
$[\{\}, \{a,b,c\}]$--r_2-->$[\{r_2\}, \{a,b,c,d\}]$
  $[\{r_2\}, \{a,b,c,d\}]$--r_1-->$[\{r_1,r_2\}, \{a,b,c,d,e\}]$
  $[\{r_2\}, \{a,b,c,d\}]$--r_4-->$[\{r_2,r_4\}, \{a,b,c,d,\neg e\}]$

$[\{\}, \{a,b,c\}]$--r_3-->$[\{r_3\}, \{a,b,c,\neg d\}]$

$[\{r_3\}, \{a,b,c,\neg d\}]$--$r_1$-->$[\{r_1,r_3\}, \{a,b,c,\neg d,e\}]$

So $Sem(0, \Pi)$
$= \cap\{B(N_d) : N \text{ is a leaf of } App(0, \Pi) \text{ and } Ds(N)\}$
$= Cn(\{a,b,c,\neg d,e\}) \cap Cn(\{a,b,c,e,d\}) \cap Cn(\{a,b,c,d,\neg e\})$
$= Cn(\{a,b,c,d \vee e\})$.
This agrees with the ambiguity propagating intuition.

A linear representation of $App(Wn, \Pi)$, with kernels, is:
$[\{\}, \{a,b,c\}]$--r_1-->$[\{r_1\}, \{a,b,c,e\}]$

So $Sem(Wn, \Pi)$
$= \cap\{B(N_d) : N \text{ is a leaf of } App(Wn, \Pi) \text{ and } Ds(N)\}$
$= Cn(\{a,b,c,e\})$.
This agrees with the ambiguity blocking intuition.
EndExample4.1

5 Default Logic

Since plausible theories encompass Default Logic, it is interesting to see how applicative digraphs relate to various default logics. We shall consider Reiter's [1980] Default Logic, and Lukaszewicz's [1990] modified Default Logic.

(Ax, D) is a default theory iff Ax is a set of sentences ($Ax \subseteq Snt$) and D is a set of defaults. Recall a default, d is a triple $(p(d) : J(d) / c(d))$, where $p(d)$ and $c(d)$ are sentences, and $J(d)$ is a non-empty finite set of sentences. Since the first-order language we are working in is countable, Snt is countable. Since $J(d)$ is finite, D is countable.

Definition 5.1 ($\Pi(\Delta)$, *plausible theory generated from* Δ).
If $\Delta = (Ax, D)$ is a default theory and Ax is satisfiable then define $\Pi(\Delta)$ as follows.
If $d = (p(d) : J(d) / c(d))$ then $\rho(d) = \{p(d)\}{:}J(d) \Rightarrow c(d)$.
Define $\Pi(\Delta) = (Ax, R, >)$, where
$R = \{\rho(d) : d \in D\}$, and $>$ is empty.

If $\Pi(\Delta) = (Ax, R, >)$ then $R = R_d$ and R_w is empty. Since $>$ is empty every node of $App(0, \Pi(\Delta))$ is desirable.

5.1 Reiter's extensions

Reiter's entailment semantics for a default theory is expressed via extensions which are usually defined as fixed points of the following function Φ.

Definition 5.2 (Φ, Reiter-extension).
Let $\Delta = (Ax, D)$ be a default theory. If F is a set of sentences define $\Phi(F)$ to be the smallest (under $\subseteq$) set satisfying E1, E2, and E3.
E1) $Ax \subseteq \Phi(F)$.
E2) $Cn(\Phi(F)) = \Phi(F)$.
E3) If $d \in D$ and $p(d) \in \Phi(F)$ and for all j in $J(d)$, $\neg j \notin F$ then $c(d) \in \Phi(F)$.
A set, E, of sentences is a *Reiter-extension* of Δ iff $\Phi(E) = E$.

Not every leaf of $App(0, \Pi(\Delta))$ corresponds to a Reiter-extension. We now characterise those that do.

Definition 5.3 (Reiter-full, Reiter-leaf).
If N is a set of rules then N is *Reiter-full* iff for all r in R, if $Ap(r, N)$ and for all g in $G(r)$ $\{g\}\cup B(N)$ is satisfiable then $r \in N$.

Let $\Delta = (Ax, D)$ be a default theory, and $\Pi(\Delta) = (Ax, R, >)$. A node, N, of $App(0, \Pi(\Delta))$ is a *Reiter-leaf* of $App(0, \Pi(\Delta))$ iff N is Reiter-full.

It can be shown that every Reiter-leaf of $App(0, \Pi(\Delta))$ is indeed a leaf of $App(0, \Pi(\Delta))$. The relationship between the Reiter-extensions of Δ and the Reiter-leaves of $App(0, \Pi(\Delta))$ is given in the following theorem.

Theorem 1
Let $\Delta = (Ax, D)$ be a default theory and Ax be satisfiable. Then E is an Reiter-extension of Δ iff there is a Reiter-leaf, N, of $App(0, \Pi(\Delta))$ such that $E = B(N)$.

5.2 Lukaszewicz's modified extensions

Pages 202 to 225 of [Lukaszewicz 1990] define and develop a default logic which has a modified definition of extension, called an m-extension. After a double fixed-point definition (page 204) of an m-extension, Lukaszewicz characterises m-extensions as (essentially) leaves of a digraph (page 218).

Definition 5.4 (Γ_1, Γ_2, m-extension).
Let $\Delta = (Ax, D)$ be a default theory. If S and U are any sets of sentences define $\Gamma_1(S, U)$ and $\Gamma_2(S, U)$ to be the smallest (under $\subseteq$) sets satisfying ME1, ME2, and ME3.
ME1) $Ax \subseteq \Gamma_1(S, U)$.
ME2) $Cn(\Gamma_1(S, U)) = \Gamma_1(S, U)$.
ME3) If $d \in D$ and $p(d) \in \Gamma_1(S, U)$ and for all j in $U \cup J(d)$, $S\cup\{j, c(d)\}$ is satisfiable then $c(d) \in \Gamma_1(S, U)$ and $J(d) \subseteq \Gamma_2(S, U)$.
A set, E, of sentences is an *m-extension* of Δ iff there is a set F such that $E = \Gamma_1(E, F)$ and $F = \Gamma_2(E, F)$.

The relationship between the m-extensions of Δ and the leaves of $App(0, \Pi(\Delta))$ is given in the following theorem.

Theorem 2
Let $\Delta = (Ax, D)$ be a default theory and Ax be satisfiable. Then E is an m-extension of Δ iff there is a leaf, N, of $App(0, \Pi(\Delta))$ such that $E = B(N)$.

6 Fixed Points

We analyse the function $\Phi(.)$ of Section 5.1 so that we may get a similar function for plausible theories.

$\Phi(F)$ cannot be constructed from F. So Reiter [1980, Theorem 2.1] defined the constructive function $\Phi(.,.)$ as follows. Suppose F and G are any sets of sentences. Define $D(G, F) =$
$\{d \in D : p(d) \in Cn(G), \text{ \& for all } j \text{ in } J(d), \neg j \notin F\}$.
Define $\Phi(F, i)$ as follows.

$$\Phi(F, 0) = Ax.$$
$$\Phi(F, i{+}1) = Cn[\Phi(F, i)] \cup c[D(\Phi(F, i), F)].$$

Then $\Phi(F) = \cup\{\Phi(F, i) : i \in \mathbb{N}\}$.

To find extensions we still have to guess F and then using $\Phi(F, i)$ check if $\Phi(F) = F$. So guidance in guessing F and a simpler constructive function would be a help in finding extensions. An equivalent version of $\Phi(.,.)$ is $\Phi'(.,.)$, defined as follows.

$$\Phi'(F, 0) = Cn(Ax).$$
$$\Phi'(F, i{+}1) = Cn(Ax \cup c[D(\Phi'(F, i), F)]).$$

It can be shown that for each i in $\mathbb{N}$,
$\Phi(F, i) \subseteq \Phi'(F, i) = Cn[\Phi(F, i)] \subseteq \Phi(F, i{+}1)$.
Hence $\cup\{\Phi'(F, i) : i \in \mathbb{N}\} = \cup\{\Phi(F, i) : i \in \mathbb{N}\} = \Phi(F)$.

Each $\Phi'(F, i)$ is semantically closed, whereas $\Phi(F, i)$ is not. Also $\Phi'(F, i)$ is simpler than $\Phi(F, i)$ because Ax is used instead of $\Phi'(F, i)$. Moreover now we can see that it is not necessary to construct a set of sentences, we only need to construct a set of defaults, say N, and then form $Cn(Ax \cup c[N])$ at the end of the construction, rather than at each stage of the construction. This greatly simplifies the constructive function.

This idea focuses attention on $D(G, F)$. The defining condition of $D(G, F)$ has two parts: an applicability condition concerning the previously constructed stage, and a consistency condition concerning the F we guessed. The first part is essential, but the second part does not have to be checked at each stage, we can just make sure that our initial guess is consistent. This provides guidance in guessing F and makes the constructive function much simpler.

Applying these ideas to the general setting of plausible theories yields the following definition.

Definition 6.1 ($\Sigma(T)$, $\Sigma(T, i)$).
Suppose $\Pi = (Ax, R, >)$ is a plausible theory. If S and T are any subsets of R and $i \in \mathbb{N}$ then define

$$\Sigma(T) = \cap\{S : \text{If } r \in T \text{ and } Ap(r, S) \text{ then } r \in S\}.$$
$$\Sigma(T, 0) = \{\}.$$
$$\Sigma(T, i{+}1) = \{r \in T : Ap(r, \Sigma(T, i))\}.$$

The following two theorems relate $\Sigma(T)$ to $\Sigma(T, i)$, and the fixed points of $\Sigma(.)$ to 0-applicative digraphs.

Theorem 3
Suppose $\Pi = (Ax, R, >)$ is a plausible theory, and $T \subseteq R$.
(1) For all i in $\mathbb{N}$,
$\Sigma(T, i) \subseteq \Sigma(T, i{+}1) \subseteq \cup\{\Sigma(T, i) : i \in \mathbb{N}\} \subseteq T$.
(2) $\Sigma(T) = \cup\{\Sigma(T, i) : i \in \mathbb{N}\} \subseteq T$.

Theorem 4
Suppose $\Pi = (Ax, R, >)$ is a plausible theory, and $N \subseteq R$. Then N is a leaf of $App(0, \Pi)$ iff $\Sigma(N) = N$ and $Cs(N)$ and $Full(0, N)$.

Corollary 5
Suppose $\Delta = (Ax, D)$ is a default theory, Ax is satisfiable, $\Pi(\Delta) = (Ax, R, >)$, and $N \subseteq R$.
(1) E is a Reiter-extension of Δ iff
$\Sigma(N) = N$, $Cs(N)$, N is Reiter-full, and $E = B(N)$.
(2) E is an m-extension of Δ iff
$\Sigma(N) = N$, $Cs(N)$, $Full(0, N)$, and $E = B(N)$.

7 Conclusions

Theorem 3 shows that the fixed points of $\Sigma(.)$ can be constructed. Theorem 4 gives a declarative characterisation of the desirable leaves of $App(0, \Pi)$. It also shows that the desired sets of rules are fixed points of $\Sigma(.)$, and gives guidance in guessing which sets of rules to check for being fixed points of $\Sigma(.)$.

Theorem 1 gives a constructive characterisation of Reiter-extensions, which is different from, but similar to, the one on page 34 of [Antoniou 1997]. Corollary 5(1) gives a new fixed-point characterisation of Reiter-extensions.

Theorem 2 gives a constructive characterisation of m-extensions. Corollary 5(2) gives a fixed-point characterisation of m-extensions. Both these characterisations are much simpler than the ones given in [Lukaszewicz 1990].

If $\Delta = (Ax, D)$ is a default theory then the leaves of $App(Wn, \Pi(\Delta))$ provides Default Logic with a new semantics which is ambiguity blocking, rather than the usual ambiguity propagating semantics.

An entailment semantics tries to capture the notion of "what follows from". But this depends on human intuition, which is different for different people. This is why several applicative digraphs have been defined. Other useful ones can be defined, but the aim was to provide an adaptable framework rather than a comprehensive one.

As indicated in the introduction, much work remains to be done to relate plausible theories and applicative digraphs to other systems of defeasible rules.

Acknowledgments

I would like to thank Torsten Schaub and Tim Cleaver for helpful discussions and comments.

References

[Antoniou, 1997] Grigoris Antoniou. *Nonmonotonic Reasoning*. MIT Press, Cambridge, Massachusetts, 1997.

[Baral, 2003] Chitta Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.

[Brewka and Eiter, 1999] Gerhard Brewka and Thomas Eiter. Preferred Answer Sets for Extended Logic Programs. *Artificial Intelligence*, 109:297-356, 1999.

[Delgrande and Schaub, 2000] James Delgrande and Torsten Schaub. Expressing Preferences in Default Logic. *Artificial Intelligence*, 123(1-2):41-87, 2000.

[Delgrande et al, 2004] James Delgrande, Torsten Schaub, Hans Tompits, and Kewen Wang. A Classification and Survey of Preference Handling Approaches in Nonmonotonic Reasoning. *Computational Intelligence*, 20(2):308-334, 2004.

[Dung, 1995] Phan Minh Dung. On the Acceptability of Arguments and Its Fundamental Role in Nonmonotonic Reasoning, Logic Programming, and *n*-person games. *Artificial Intelligence*, 77(2):321-357, 1995.

[Lukaszewicz, 1990] Witold Lukaszewicz. *Non-Monotonic Reasoning Formalization of Commonsense Reasoning*. Ellis Horwood, 1990.

[Nute, 1994] Donald Nute. Defeasible Logic. In D.Gabbay, C.Hogger, and J.Robinson, eds *Handbook of Logic in Artificial Intelligence and Logic Programming*, 3:353-395, Oxford University Press, 1994.

[Reiter, 1980] Raymond Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13:81-132, 1980.