Centrum voor Wiskunde en Informatica
REPORT*RAPPORT*

# Suggestions for a non-monotonic feature logic

W.C. Rounds and G.-Q. Zhang

Computer Science/Department of Software Technology

**CS-R9551 1995**

Report CS-R9551
ISSN 0169-118X

CWI
P.O. Box 94079
1090 GB  Amsterdam
The Netherlands

# Suggestions for a Non-monotonic Feature Logic

William C. Rounds
*CWI/ University of Michigan*
*P.O. Box 94079/ Artificial Intelligence Laboratory*
*1090 GB Amsterdam/ Ann Arbor, Michigan*
*Netherlands/ USA*

Guo-Qiang Zhang
*Department of Computer Science*
*University of Georgia*
*Athens, Georgia 30602*
*USA*

## Abstract

We use Scott's domain theory and methods from Reiter's default logic to suggest some ways of modelling default constraints in feature logic. We show how default feature rules, derived from default constraints, can be used to give ways to augment strict feature structures with default information.

*CR Subject Classification (1991):* F.3, F.4, I.2
*Keywords & Phrases:* defaults, domains, non-monotonicity, feature logic

## 1. Introduction

This paper is a mathematical treatment of some of the issues which have arisen in trying to define a version of nonmonotonic feature logic which would adequately reflect the semantics of strict and default feature constraints as in the theory of Generalized Phrase Structure Grammar (GPSG) [5], and to a lesser extent, in Head-driven Phrase-Structure Grammmar (HPSG) [18].

The problem seems fairly straightforward on the surface, since there are many tools available from the non-monotonic reasoning literature. In particular, one expects that a version of Reiter‘s default logic [21], tailored to the case of feature logic, would be the appropriate tool to use, especially since feature logic can be translated into first-order language [9]. We have found, though, that such a translation does not get at the issues involved. One problem with this is that a straightforward use of default rules in default logic would model default implicative constraints using default rules, but strict constraints using material implication. This means that one cannot compare strict constraints with default constraints in an adequate way. There is in addition the well-known inability of default logic to be able to “reason by cases”. This reoccurs in our setting of strict and default constraints.

Our solution to these limitation problem is to use domain theory, or the theory of complete partial orders. We follow Pollard and Moshier [17] in modelling disjunctive information using so-called *powerdomains*. We specifically use the *Smyth* powerdomain to add disjunctive information to feature structures. This is like using default logic, but we depart from standard default logic by using a nonstandard kind of default. Our construction resembles that of Gelfond *et al.* [6], who define a notion of “disjunctive default.” Our semantics, though, is not really the same, as we work in the domain-theoretic setting, while the Gelfond setting is still that of first-order logic. Our crucial definition comes from the domain-theoretic semantics of data bases, due to Libkin [14]. Using the idea of *or-sets*, and *updates*, we are able to find a natural notion of default which allows us to characterize the notion of “completing” a feature structure with both strict and default constraints.

Our basic data structure is the *typed feature structure*, discussed in Carpenter’s book [3]. We start with *recursive* type constraint systems, as in [3, Chapter 15]. Our setting is slightly different from that used by Carpenter, but we follow his ideas in defining the notion of *resolved* or “completed” feature structure. We show, though, how to characterize (minimal) resolved structures using *default* information: this treats what might seem to be a *monotonic* construction with a *non-monotonic* tool. The difficulty resides in defining the right non-monotonic tool, and this is what we have done with updates and the Smyth powerdomain.

At the end of the paper we propose a definition of non-monotonic feature logic using our new default semantics. We define a notion of non-monotonic entailment between formulas of feature logic; for a sample feature logic we propose a logic with negation due to Dawar and Vijayshanker [4], but using a typed feature signature as in Carpenter’s book. Such a logic is able to express most of the constraints commonly found in GPSG or HPSG-like theories, including constraints on coreference. We briefly indicate how our entailment notion might or might not satisfy the laws due to Kraus, Lehmann, and Magidor [12].

We also suggest how to incorporate these notions into inheritance hierarchies. This is mostly a theoretical solution, though, as we suggest a priority scheme for building default extensions in a “layered” fashion, starting with defaults associated with the most specific levels of the hierarchy, and then making “extensions of extensions.” This is of course computationally inefficient if not impossible, but it does suggest a way of thinking about the “specificity” problem, which is difficult to do in the ordinary setting of default logic.

Young’s thesis [26] treats some of these issues: he considers disjunctive constraints, inheritance, and strict and default information using the method of *nonmonotonic sorts.* He gives attention to *prioritized* systems of defaults in which more specific default rules are given preference to less specific ones (associated with general levels in the inheritance hierarchy.) He does not work out the semantics of resolved structures in full, though he does consider the idea. For us, the full semantics is a considerable complication, especially given Blackburn and Spaan's result that in general, the existence of a resolved structure given an arbitrary constraint system is undecidable [1]. For this reason, we will generally ignore algorithms for computing resolved structures and extensions.

Lascarides, Briscoe, Asher, and Copestake [13] have also treated these problems. They extend the work of Young and Rounds [25], and some of the work in Young's thesis, by introducing a notion of order-independent and persistent typed default unification. There is actually a family of such unification operations, each of which makes the important distinction between strict and default information. In the present paper, though, we do not consider such operations; instead we focus on the semantics of the constraints, in the hope that these unification operations, which are crucial to computational realizations, will be seen as ways to implement the constraints, much as the ordinary notion of unification is a way of implementing the logical operation of conjunction.

The plan of the paper is this. We first cover basics of default domain theory in Section 2. Then we review feature structures and logic in Section 3, and in Section 4 review previous work on constraints. Section 5 is a warm-up for the results in Section 6. It consists of rephrasing some of Carpenter's work on constraint resolution; we treat only the case of conjunctive resolution here, but provide our first characterization of "resolution as default extension". Then in Section 6 we define the Smyth powerdomain, antichain updates, and prove the general disjunctive default characterization of resolution. Section 7 talks about non-monotonic entailment, and Section 8 returns to a specific logic, together with the notion of inheritance.

## 2. Basic default domain theory

Here we give a short review of basic domain theory, and a presentation of defaults in this setting.

### *2.1 Scott domains*

We present a list of the most important definitions.

- A complete partial order (cpo) is a poset $(D, \sqsubseteq)$ with a bottom element and least upper bounds of directed sets. (A set $X$ is directed if it is nonempty, and for any two elements $x, y$ in $X$ there is $z$ in X with $x, y$ both $\sqsubseteq z$.)
- A subset $X \subseteq D$ is consistent if it has an upper bound in $D$.
- A compact element $x$ of $D$ is one such that whenever $x \sqsubseteq \bigsqcup X$ with $X$ directed, we also have $x \sqsubseteq y$ for some $y \in X$. The set of compact elements of a cpo $D$ is written as $\kappa(D)$.
- A cpo is algebraic if each element is the least upper bound of a set of finite elements.
- A cpo is $\omega$-algebraic if it is algebraic and the set of finite elements is countable.
- A Scott domain is an $\omega$-algebraic cpo in which every consistent subset has a least upper bound.
- We write $x \uparrow y$ if the set $\{x, y\}$ is consistent.
- $\uparrow x = \{y \mid y \sqsupseteq x\}$.

Now for some examples:

- **A domain for birds:** Let $D$ be the set

$$\{\bot, bird, fly, tweet, peng, ftweet, pengtweet\}$$

with $bird \sqsubseteq tweet$; $bird \sqsubseteq peng$; $tweet \sqsubseteq ftweet$; $tweet \sqsubseteq pengtweet$; and $peng \sqsubseteq pengtweet$. The relation $\sqsubseteq$ is the smallest with $\bot \sqsubseteq x$ for all $x$ and containing the above pairs. Being a penguin is inconsistent with flying; being named tweety is consistent with flying and being a penguin; all things named tweety are birds; and penguins are birds. This domain is a familiar example of an inheritance hierarchy.

- **The complete binary tree:** Let $D = \Sigma^* \cup \Sigma^\omega$, where $\Sigma = \{0, 1\}$. This is the domain of finite and infinite binary strings, with $\sqsubseteq$ being the prefix relation.
- **The Scott domain of closed first-order theories:** Let a proof system for first-order logic be given. Take $D$ to be the collection of all deductively closed consistent sets $T$ of sentences, ordered by inclusion. The bottom element of this domain is the set of logically valid sentences; the finite elements are the theories generated by finite sets (in fact singleton sets) of formulas.

The next example: the *domain of feature structures*, forms the major application area of the paper. For this the relevant linguistic theories are *feature-based* theories like HPSG [18]. We focus on the notion of *typed* feature structures [3], which are the bearers of linguistic information in such theories[1].

**Definition 2.1** *Let $L$ be a set of* **feature labels** *and $A$ be a finite Scott domain of types. An* **abstract typed feature structure** *is a triple $(T, N, \theta)$, where $T$ a prefix-closed nonempty subset of $L^*$, $\theta$ maps $T$ to $A$, and $N$ is a right-invariant equivalence (Nerode) relation on $T$ respecting $\theta$; that is, if $p$ and $q$ are strings in $T$, with $p\ N\ q$, and $pf \in T$ for a feature $f$, then (i) $qf \in T$ and $pf\ N\ qf$ (similarly if $qf \in T$); (ii) if $p\ N\ q$ and $\theta(p) = a$ then $\theta(q) = a$.*

This rather imposing definition (due originally to Moshier [15], and anticipated by Pereira and Shieber [16]) captures notations like the following:

$$\begin{bmatrix} \text{vp} & & \\ \text{AGR} & \boxed{1} & \begin{bmatrix} \text{NUM} & \text{sing} \\ \text{PERS} & \text{3rd} \end{bmatrix} \\ \text{SUBJ} & \boxed{1} & \end{bmatrix}$$

Here the feature names are things like AGR and NUM, whilst types are things like "vp" and "3rd". The set $P$ in this example would contain "paths" such as AGR NUM and AGR PERS. The function $\theta$ assigns "vp" to the empty path, and "sing" to AGR NUM. The box $\boxed{1}$ indicates coreference, or "structure-sharing", and is represented by the Nerode relation; so that SUBJ is N-related to AGR (consequently there is really a "hidden path" SUBJ NUM in the above structure.)

**Notation.** We will have occasion for several examples, so we show a linear way of presenting the above structure:

$$vp[agr : \boxed{1}\ [num : sing, pers : 3rd], subj : \boxed{1}\ ].$$

An immediate consequence of the above definition is that the collection of typed feature structures forms a Scott domain under the natural componentwise ordering (inclusion of the $T$ and $N$ components, and pointwise ordering of the $\theta$ functions.) We say that the feature structure $F$ is of type $a$ if $\theta(\epsilon) \sqsupseteq a$ in the domain $A$.

We could consider other restrictions on the domain of feature structures imposed by Carpenter [3]. Structures could be required to obey *appropriateness conditions*; and we dcould also require structures to include *disjointness* information, essentially barring the possibility of structure-sharing. All of these restrictions can be incorporated into our theory, since the only real conditions needed for our results are general ideas definable in any Scott domain.

[1] The current concern of HPSG, however, is not on the partiality of information expressible with feature structures, but on feature structures as descriptions of linguistic entities all of whose properties are determined.

*2.2 Defaults*

Having the basic domain-theoretic definitions in hand, let us turn to the study of default information. Our intention is to apply the general theory in the case of the domain of feature structures, but we begin with the general case.

**Definition 2.2** *Let $(D, \sqsubseteq)$ be a Scott domain. A default set is a subset $\Delta$ of $\kappa(D) \times \kappa(D)$. We call a pair $(a, b) \in \Delta$ a (normal) default and think of it as a rule $\dfrac{a : b}{b}$.*

A rule like $\dfrac{a : b}{b}$ intuitively means that if $a$ is part of a current state and $b$ is compatible with this state, then $b$ can be added to the state, where "state" is modeled by a domain element. Of course this is very vague, and indeed there are several different ways to make this intuition precise. The general sense is that if $\Delta$ is a default set, and $x$ is an element in $D$, then we can use $\Delta$ to get to an element $y \sqsupseteq x$, containing more information than $x$. Therefore, from an abstract point of view, a default set in a Scott domain $(D, \sqsubseteq)$ serves to generate a certain relation $\Upsilon$ on $D$ which at least satisfies the property that $(x, y) \in \Upsilon$ implies $x \sqsubseteq y$. We take $\Upsilon$ to be the *extension relation*, due to Reiter [21] in the case where the domain $D$ is the domain of closed first-order theories.

**Definition 2.3** *Let $(D, \sqsubseteq)$ be a Scott domain. Let $\Delta$ be a default set in $D$. Also, define $D^\top$ to be the domain $D$ with an "inconsistent element" $\top$ above all other elements. It is well-known, and easy to prove, that $D^\top$ becomes a complete lattice. Now define the function $\phi(x, y)$ from $D \times D$ to $D^\top$ as follows: $\phi(x, y) = \bigsqcup_i \phi(x, y, i)$, where $\phi(x, y, 0) = x$, and*

$$\phi(x, y, i + 1) = \phi(x, y, i) \sqcup \bigsqcup \{b \mid (a, b) \in \Delta \ \& \ a \sqsubseteq \phi(x, y, i) \ \& \ b \uparrow y\}$$

*for all $i \geq 0$.*

*We write $x \Upsilon y$ if $x$ and $y$ are both in $D$ and $\phi(x, y) = y$. When $x \Upsilon y$, we call $y$ an extension of $x$.*

**Remark.** It is important to note that the use of $D^\top$ in the above definition is only to ensure that the construction of extensions at each stage makes sense. We would not consider the added "top" element to be an extension, even though it might be a fixed point of the function $\lambda y.\phi(x, y)$. This point is raised again in the sections on the Smyth powerdomain.

Here is an example from [27] to illustrate the basic idea of the definition.

**Example**. Consider the scenario of finding out somebody's last name if we have the partial information that the name starts with 'sm'. Although we only have partial information, it would be a good guess if we say that the last name is 'smith'. Thus, the pair ($\texttt{sm}, \texttt{smith}$) is a good candidate for a default rule ( `smyth` would be an exception).

To be more specific, consider the complete binary tree, where the elements are binary strings of finite and infinite length, so that $w \sqsubseteq v$ if and only if $w$ is a prefix of $v$. Let the default set be

$$\Delta := \{(w11, w111) \mid w \text{ is a finite binary string}\}.$$

Intuitively, the defaults say that if we see two consecutive 1's, then mostly likely we will see another 1. The following are sample pairs in the extension relation:

$$(1111, 11111), (1101, 1101), (1^\omega, 1^\omega).$$

□

Clearly, if $\Delta$ is empty or the identity relation, then the extension relation is the identity relation. Also, for maximal elements $m$, like $1^\omega$ above, we always have $m \Upsilon m$. This matches our intuition: if we already have perfect information about some object, defaults can tell us nothing more about the object.

Note that although an extension is a certain fixed point, the definition only provides a way to confirm one rather than to find one. An extension seems to build up in stages, but at each step certain consistency with the extension must be checked. This is anomalous: to construct an extension, we must already know it! It is also this phenomenon that makes the extension relation nonmonotonic: if $y$ is an extension of $x$ and $x' \sqsupseteq x$, then $y$ need not be an extension of $x'$.

Do extensions always exist? What kind of properties does the extension relation have? The following theorem summarizes some important properties of extensions for default domains.

**Theorem 2.1** *Given a Scott domain $D$ and a default set $\Delta$, we have:*

1. *Extensions always exist.*
2. *If $x \Upsilon y$ then $y \sqsupseteq x$.*
3. *$x \Upsilon y$ and $y \Upsilon z$ if and only if $y = z$.*
4. *If $x \Upsilon y$ and $x \Upsilon y'$, then either $y = y'$ or $y \not\uparrow y'$.*
5. *If $x \Upsilon z$ and $y \sqsubseteq z$, then $(x \sqcup y) \Upsilon z$.*

**Proof**. The original proofs for these, in terms of information systems (a representation of Scott domains), are given in [23]. For completeness, we give the proof of the existence of extensions, exactly like the proof of Reiter, but generalized to domains.

We define a sequence $\langle x_i \rangle$ inductively. Let $x_0 = x$; and for each $i > 0$, let $U_i$ be a subset of $D$ which is maximal among subsets $U$ of

$$\{b \mid \exists a((a, b) \in \Delta \ \&\ a \sqsubseteq x_{i-1})\}$$

with the property that

$$\{x_i\} \cup U \text{ is bounded.}$$

Put $x_i = x_{i-1} \sqcup \bigsqcup U_i$. It is trivial that $\langle x_i \rangle$ is an increasing chain. Let

$$M := \bigsqcup_{i \in \omega} x_i.$$

We show that

$$M = \bigsqcup_{i \in \omega} \phi(x, M, i).$$

First we claim

$$U_i = \{b \mid \exists a((a, b) \in \Delta \ \&\ a \sqsubseteq x_{i-1} \ \&\ b \uparrow M)\}.$$

Clearly any $b \in U_i$ is also included in the right side, because $M$ is the least upper bound of all the $x_i$. If $b$ is in the right side but not the left, then by maximality of $U_i$, we have that $\{x_{i-1}\} \cup U_i \cup \{b\}$ is inconsistent. This implies that $M \sqsupseteq (x_{i-1} \sqcup \bigsqcup U_i)$ is inconsistent with $b$, contradicting the fact that $b$ is in the right side. This establishes the claim; then by induction we obtain

$$x_i = \phi(x, M, i),$$

the inductive step using the claim just proved. The result follows immediately.

□

Before answering these questions, we present a characterization of extensions. It generalizes an early result of Reiter's.

**Theorem 2.2** *For a Scott domain $(D, \sqsubseteq)$ and a subset $\Delta \subseteq \kappa(D) \times \kappa(D)$, we have $x \Upsilon y$ if and only if*

$$y = \bigsqcap \{\, t \ \mid \ t = x \sqcup \bigsqcup \{b \ \mid \ (a,b) \in \Delta \ \& \ a \sqsubseteq t \ \& \ b \uparrow y\} \,\}.$$

This theorem suggests that extensions can be characterized as a nesting of least fixed point and greatest fixed point. The proof can be found in [22]. The point of the theorems is that techniques originally developed for default logic can be generalized to many more situations.

The following paragraphs provide a whole class of examples, this time involving a monotonic notion. We give these because our main theorems have a similar flavor. We recall the definition of a continuous function from a domain $D$ to itself.

**Definition 2.4** *A function $f : D \to D$ is said to be* continuous *iff for any directed set $X \subseteq D$,*

$$f(\bigsqcup X) = \bigsqcup \{f(x) \mid x \in X\}.$$

Well-known facts about continuous functions are the following:

**Lemma 2.1 (Folklore.)**

- *If $f$ is continuous, then it has a least fixed point*

$$d_0 = \bigsqcup_{i \in \omega} f^i(\bot);$$

- *For any $d \in D$,*

$$f(d) = \bigsqcup \{b \in \kappa(D) \mid (\exists a \in \kappa(D))(a \sqsubseteq d \text{ and } b \sqsubseteq f(a)\}.$$

We characterize least fixed points using extensions. Assume given a Scott domain $D$ and continuous $f : D \to D$. For such a function $f$, let the set of defaults $\Delta(f)$ be

$$\{(a,b) \in \kappa(D) \times \kappa(D) \mid b \sqsubseteq f(a)\}.$$

**Proposition 2.1** *The least fixpoint $d_0$ of $f$ is an extension of $\bot$ with respect to $\Delta(f)$.*

**Proof.** Consider the functions $\phi(x, y, i)$ used to define extensions. We show by induction on $i$ that

$$\phi(\bot, d_0, i) = \bigsqcup_{j=0}^{i} f^j(\bot).$$

The basis is obvious. Assume the result for $i$. Put $f_i = \bigsqcup_{j=0}^{i} f^j(\bot)$. Then

$$\phi(\bot, d_0, i+1) = f_i \sqcup \bigsqcup \{b \mid (\exists a)(a \sqsubseteq f_i \wedge b \sqsubseteq f(a) \wedge b \uparrow d_0)\}.$$

However, if $b$ is one of the elements on the right, if $a \sqsubseteq f_i$ then $f(a) \sqsubseteq f(f_i) \sqsubseteq d_0$. Therefore if $b \sqsubseteq f(a)$ then $b \sqsubseteq d_0$, so the clause $b \uparrow d_0$ can be dropped. Apply Lemma 2.1 with $d = f_i$. Then we get

$$\phi(\bot, d_0, i+1) = f_i \sqcup f(f_i) = f_{i+1}.$$

□

## 3. Feature logic

The language **KR** was introduced in [10], [11] to solve the problem of expressing disjunctive information in feature structures, while at the same time capturing the constraints implicit in Shieber's PATR-II [24]. The language **KR** consists of *basic* and *compound* formulas. Assuming $L$ and $A$ as above, the basic formulas of **KR** are as follows

- (Constants) $a$ for each $a \in A$;
- (Truth) The special formulas **true** and **false**,
- (Path equations) $p \doteq q$ for $p, q \in L^*$.

Then the compound formulas are given inductively. If $\varphi$ and $\psi$ are formulas, then so are

- $\varphi \wedge \psi$;
- $\varphi \vee \psi$;
- $l : \varphi$ for $l \in L$.

The semantics of Kasper-Rounds logic is straightforward given the concept of an abstract feature structure. We need only one preliminary definition. Let $F = (T, N, \theta)$ be an abstract feature structure, and $p$ be a path of $F$. Then $F/p$ ($F$ after $p$) is the structure $(T/p, N/p, \theta_p)$, where $T/p = \{r \mid pr \in T\}$; $N/p$ holds between two paths $r$ and $q$ iff $N$ holds between $pr$ and $qr$, and $\theta_p(r) = \theta(pr)$.

The semantics of formulae is now the usual one. Let $F$ be a feature structure.

- $F \models a$ if $\theta(F/\epsilon) \sqsupseteq a$ ($\epsilon$ is the null path);
- $F \models$ **true** always and **false** never;
- $F \models p \doteq q$ if $F/p = F/q$ (note that this requires that the paths $p$ and $q$ are in the path set of $F$);
- $F \models \varphi \wedge \psi$ if $F \models \varphi$ and $F \models \psi$;
- $F \models \varphi \vee \psi$ if $F \models \varphi$ or $F \models \psi$;
- $F \models l : \phi$ if $F/l \models \varphi$.

Any formula in Kasper-Rounds logic has a finite number of subsumption-minimal satisfying abstract feature structures. The basic fact about the logic is then that if $F$ is a satisfier of the formula $\varphi$, and $G$ of $\psi$, then the unification of $F$ and $G$ is a satisfier of the conjunction $\varphi \wedge \psi$, assuming that the conjunction is satisfiable. This fact makes it possible to compute the minimal satisfiers of the conjunction of two disjunctive formulas by pairwise unifying the minimal satisfiers of the two separate formulas, and thus leads to an elucidation of the "semantics of disjunctive feature structures" by regarding a disjunctive structure not as a semantic entity, but instead, a disjunctive formula – a description of a set of non-disjunctive structures. A consequence of this is that we can pass back and forth between formulas and the set of minimal satisfiers of these formulas. (See Carpenter [3] for full information on how to do this.)

One other distinguishing characteristic of Kasper-Rounds logic is *persistence.* Because the logic contains no negations, we have the fact that if an element $d$ satisfies $\varphi$, and $d \sqsubseteq e$, then $e$ satisfies $\varphi$ too. Combined with the results in the previous paragraph, this means that the set of satisfiers of any Kasper-Rounds formula can be described as the upward closure, in the subsumption preorder, of the set of minimal satisfiers.

## 4. Strict and default constraints

### *4.1 Strict constraints: FCRs*

In the linguistic theory HPSG [18], feature structures must be *constrained.* One formalization of this idea is that every substructure of a feature structure which is of a certain type (in $A$) must satisfy a **KR** formula which has been associated with that type beforehand. The formula is obtained by first mapping each separate type to an "initial" formula; then the final formula associated with a type is the conjunction of all the formulas which have been mapped initially to more general types. Thus a type inherits constraints from all of its supertypes. Carpenter formalizes this idea as *recursive type constraint systems* [3, Chapter 15]. We begin with an example not involving inheritance.

**Example 4.1** *Suppose that ceo is a type, and consider the expression*

$$\textit{employee} \Rightarrow \textit{ceo} \vee \textit{boss: employee.}$$

Intuitively, this expresses a structured type, whose members are all those persons who have some chain of supervisors leading up to the chief executive officer. But we might also wish the constraint to be satisfied by the cyclic feature structure consisting of one node of type *employee*, with a single attribute *employer* pointing back to that same node. This would be a person who is self-employed.

Carpenter gives a general way to formalize these ideas, which allows such cyclic structures as solutions: Let **KR** be the set of Kasper-Rounds formulae, and $A$ be the set of sort symbols. A *constraint system* is then a mapping $C : A \to \mathbf{KR}$. To deal with inheritance correctly, we then extend $C$ as follows:

$$C^*(a) = \bigwedge_{b \sqsubseteq a} C(b).$$

Now let $F = (T, N, \theta)$ be a typed feature structure. $F$ is said to be *resolved with respect to $C$* iff for all $p \in T$,

$$F/p \models C^*(\theta(p)).$$

We wish to present an alternative definition here: one which will allow a slightly more general type constraint system, similar to one appearing in the linguistic theory Generalized Phrase-Structure Grammar (GPSG) [5]. That theory allows constraints on structures like the following:

$$[+\mathrm{INV},\ \mathrm{BAR}\ 2] \supset [\mathrm{SUBJ}].$$

These constraints are called Feature Co-occurrence Restrictions (FCRs). The example constraint says of a feature structure that if it represents the type of a verb which can introduce an inverted clause, then it has a subject (of some type.) Again we wish this constraint to hold at every substructure of a given feature structure.

Leaving aside the question of inheritance for the moment, we can formalize such constraints as "rules" of the form

$$\phi \to \psi$$

where $\phi$ and $\psi$ are KR formulas. Initially we will take both formulas to be disjunction-free. Notice that we do not consider "$\rightarrow$" to be a logical connective, though its force is exactly that of classical material implication. Given a set $C$ of these rules, we can then say a feature structure $F$ is $C$-resolved iff for each path $p \in F$, and each $(\phi \rightarrow \psi) \in C$, if $F/p \models \phi$ then $F/p \models \psi$.

One can express Carpenter's typing notion in this framework as follows. For each type $a$ we introduce the rules

$$(a \rightarrow C^*(a))$$

for each atomic type $a$. We could also consider a variant notion in which for each type $a$, there was a set $C(a)$ of GPSG-style constraints, and then require inheritance of these onto subtypes.

Blackburn and Spaan [1], and Reape [20] have investigated this direction extensively. In their formulation, the type symbols $a$ can be thought of as special propositional variables, perhaps rewritten as $p_a$. The modality $l : \varphi$ is written $\langle l \rangle \varphi$, and a negation operator $\neg\phi$ is included. The use of the $\langle\ \rangle$ notation is derived from the $\diamond$ or possibility modality in standard modal logic.

A feature structure $F$ can be rechristened a *Kripke model*, or *polyframe* in modal logic terminology. Now the paths $p$ of the structure are thought of as possible worlds, and the interpretation of the arc labels $l$ as unary partial functions give several accessibility relations between the worlds. Blackburn and Spaan investigate various extensions of this modal formalism; one of these is the *universal modality* $\Box$. Let $\varphi$ be a formula of KR. Then the semantics of $\Box\varphi$ is simply described as follows. Fix a feature structure $F$. Then

$$F \models \Box\varphi \iff (\forall p \in T(F))\,(F/p \models \varphi).$$

This definition could be adapted to our situation were we to allow for classical implication as a logical connective. But for reasons which will become a bit clearer as we go on, we do not want to take this route.

## *4.2 Default restrictions: FSDs*

GPSG makes another kind of co-occurrence restriction on feature structures: *feature specification defaults*, or FSDs. These have exactly the same form as FCRs, but are considered to be rules whose effects can be over-ridden by strict constraints; only in the absence of a strict constraint with contradictory force can a FSD be assumed to hold. A simple example is

$$\text{[-INV]}$$

which asserts that normally a structure is of the type of a non-invertive verb. This could be phrased in our terms as the "rule"

$$\textbf{true} \rightarrow inv : -,$$

where **true** is the "always true" KR formula.

Our challenge is now to make sense of such default constraints in the framework of default model theory. We have to satisfy several criteria:

- The mathematical theory must conform to the linguistic one insofar as possible.
- We have to account both for strict and default constraints, in a way which clarifies their similarities and differences.
- We need to respect the "universal modality" idea implicit in resolved structures.
- Our theory should be sensitive to the way FCRs and FSDs fall in an inheritance hierarchy.
- Eventually we need to account for constraints with disjunctions.

## 5. Some semantics for FCRs

We start with a simple situation in which all constraints are of the form $(\phi \to \psi)$ with $\phi$ and $\psi$ conjunctive. We also begin with a feature structure $F$ and consider how to construct a minimal resolved feature structure $\overline{F}$ with $F \sqsubseteq \overline{F}$. Of course there may be no such structure; but we show that if there is one, then there is a least such in the subsumption order. Further, in analogy with a result of Carpenter [3, Chapter 15], this structure (if it is finite) may be found by using the FCR's as rewriting rules.

We then go on to show that one may also understand the situation by regarding FCRs as *default rules* in the domain of feature structures. This comes about because a pair $(\phi, \psi)$ of conjunctive formulas may as well be regarded as a pair $(G, G')$ of feature structures: simply take minimum satisfiers, respectively, of the two formulae. Then we use the pairs $(G, G')$ as default rules. Once we have augmented these rules to account for the universal modality property, we will be in a position to show that if there is a minimum resolved feature structure $\overline{F}$ subsumed by our given $F$, then $\overline{F}$ is the unique *default* extension of $F$ using the default rules (but not conversely.)

Now we begin with some details. Assume given a finite set $S$ of pairs of the form $(\phi, \psi)$ with $\phi$ and $\psi$ conjunctive formulas of **KR** logic. We could in fact assume that $S$ was a (partial) function, since the existence of pairs $(\phi, \psi)$ and $(\phi, \theta)$ in $S$ is taken to mean that both constraints hold, so that we could replace these pairs by $(\phi, \psi \wedge \theta)$. However, in the default case it might be that $\psi$ and $\theta$ were inconsistent, and even if they were consistent, there could be interactions with other pairs. So we keep $S$ as a set of pairs for future comparison with the default setting.

For each pair $(\phi, \psi)$, construct the pair $C = (C_1, C_2)$ where $C_1$ and $C_2$ are the least satisfiers of $\phi$ and $\psi$ respectively. We henceforth regard $S$ as a set of such pairs.

**Definition 5.1** *For a path $p \in L^*$, and feature structure $G = (T, N, \theta)$, the structure $pG$ is defined to be the structure*

$$(pT, pN, p\theta)$$

*where $pT$ is the prefix-closure of $\{pt \mid t \in T\}$, $pN$ is the identity on all prefixes of $p$ and $pq$ $(pN)$ $pr$ iff $q$ $N$ $r$; and $p\theta(s) = \perp$ whenever $s$ is a prefix of $p$, and otherwise $p\theta(pq) = \theta(q)$. The structure $pG$ is called the* translation *of $G$ along $p$ (cf. Pollard and Moshier [17]).*

We make the following observation, whose proof is straightforward.

**Lemma 5.1** *For a feature structure $F$ with path $p \in T(F)$, and feature structure $G$,*

$$G \sqsubseteq F/p \iff pG \sqsubseteq F.$$

Now we want to use the set $S$ of constraints as rewriting rules for feature structures.

**Definition 5.2** *Let $F, F'$ be feature structures and $C = (C_1, C_2)$ be a constraint in $S$; further let $p$ be an arbitrary string of feature names. We say that $F \overset{p,C}{\Rightarrow} F'$ iff $p \in T(F)$ and either (i) $pC_1 \not\sqsubseteq F$ and $F = F'$, or (ii) $pC_1 \sqsubseteq F$ and $F' = F \sqcup pC_2$.*

This definition gives rise to our next observation, which is very much the same as one of Carpenter's:

**Lemma 5.2** *Let $S$ be a finite set of constraints. A feature structure $F$ is resolved with respect to $S$ iff $F \overset{p,C}{\Rightarrow} F$ for all constraints $C \in S$, and all paths $p$ of $F$.*

**Proof.** Straightforward with the definitions and previous lemma.

□

Finally, let $F$ be a feature structure; then define $\overline{F}$ to be the least feature structure $G$ such that $G$ is resolved (with respect to $S$) and $F \sqsubseteq G$, if there is such. We must prove the existence of $\overline{F}$.

**Lemma 5.3** *Let $F$ be a feature structure and suppose that there is a $G$ such that $G$ is resolved and $F \sqsubseteq G$. Then the generalization $\overline{F}$ of all such $G$ is resolved.*

(The generalization of a set of feature structures is its greatest lower bound in the subsumption order.)

**Proof.** Let $p$ be a path of $\overline{F}$ and let $C = (C_1, C_2)$ be a constraint in $S$. We need to show that $\overline{F} \overset{p,C}{\Rightarrow} \overline{F}$. Assume that $pC_1 \sqsubseteq \overline{F}$. Then this holds for all $G$ with $\overline{F} \sqsubseteq G$. So if $G$ is in fact resolved, we have $C_2 \sqsubseteq G/p$, and so

$$C_2 \sqsubseteq \sqcap\{G/p \mid G \text{ is resolved}, F \sqsubseteq G\},$$

which is the same as saying $C_2 \sqsubseteq \overline{F}/p$.

□

We want to show that $\overline{F}$ can be reached "from below" by some (finite or infinite) sequence of rewritings. For this purpose, we introduce a " parallel" rewriting relation. as in Carpenter [3, Definition 196].

**Definition 5.3** *Assume given a finite set $S$ of constraints of the form $(C_1, C_2)$. Let $F$ and $G$ be feature structures. We say that $F \overset{S,par}{\Rightarrow} G$ iff*

$$G = F \sqcup \bigsqcup\{pC_2 \mid (C_1, C_2) \in S, pC_1 \sqsubseteq F\}.$$

Notice first that this definition determines $G$ functionally once $F$ is given. We sometimes write $G = S(F)$ when $F \overset{S,par}{\Rightarrow} G$. Notice second the developing resemblance to the clauses in the definition of extensions (Definition 2.3.) In particular the least upper bound in our definition may in general have to be taken with respect to the domain of feature structures with a top element denoting an inconsistent structure; but we will fortunately be able to avoid uses of the definition where this inconsistency manifests itself. We have a lemma analogous to Lemma 5.2. The proof is easy.

**Lemma 5.4** *A feature structure $F$ is resolved with respect to $S$ iff $F \overset{S,par}{\Rightarrow} F$.*

□

Before stating the next results, we recall that a feature structure is *finite* iff its Nerode relation is of finite index. The finite feature structures are in fact the compact elements of the domain of abstract feature structures.

The following lemma is again attributable to Carpenter [3, Theorem 192]. However, in this lemma we are using breadth-first rewriting instead of the rewriting used by Carpenter.

**Lemma 5.5** *Let $F$ be a finite feature structure, and let $\overset{S,par^*}{\Rightarrow}$ be the reflexive, transitive closure of the relation $\overset{S,par}{\Rightarrow}$. If $\overline{F}$ exists and is finite then $F \overset{S,par^*}{\Rightarrow} \overline{F}$.*

**Proof.** Construct a sequence $F_i$ of feature structures with $F_i \sqsubseteq F$ for each $i$ as follows. Let $F_0 = F$, and $F_{i+1} = S(F)$. Because $F_i \sqsubseteq \overline{F}$ inductively, whenever $pC_1 \sqsubseteq F_i$ we have $pC_1 \sqsubseteq \overline{F}$. Since $\overline{F}$ is resolved, the corresponding $pC_2 \sqsubseteq \overline{F}$. Thus $S(F) = F_{i+1} \sqsubseteq \overline{F}$.

Now we claim that the least upper bound $\bigsqcup_i F_i$ of this sequence is resolved. For if $C = (C_1, C_2)$ is a constraint, and $pC_1 \sqsubseteq \bigsqcup F_i$, then since $pC_1$ is compact, we have that for some $j$, $pC_1 \sqsubseteq F_j$, and so $pC_2 \sqsubseteq F_{j+1}$ by construction.

It follows from the last two paragraphs that $\bigsqcup_i F_i = \overline{F}$. We assumed, though, that $\overline{F}$ was finite (compact); so in fact $\overline{F}$ is one of the $F_j$, and thus $F \overset{S,par^*}{\Rightarrow} \overline{F}$.

□

The previous lemma puts us in a position to prove the same result, but using the "sequential" rewriting relation introduced (in essence) by Carpenter. Define for $S$ a set of constraints

$$\overset{S,seq}{\Rightarrow} = \bigcup\{ \overset{p,C}{\Rightarrow} \mid p \in L^*, C \in S\}.$$

**Theorem 5.1 (Carpenter)** *Let* $\overset{S,seq^*}{\Rightarrow}$ *be the reflexive, transitive closure of* $\overset{S,seq}{\Rightarrow}$*, and let $F$ be a finite feature structure. If $\overline{F}$ exists and is finite then* $F \overset{S,seq^*}{\Rightarrow} \overline{F}$*.*

**Proof.** We know that $F \overset{S,par}{\Rightarrow}{}^* \overline{F}$. Consider the definition of $\overset{S,par}{\Rightarrow}$. The term

$$\bigsqcup\{pC_2 \mid pC_1 \sqsubseteq F, (C_1, C_2) \in S\}$$

takes a least upper bound over a possibly infinite set of feature structures. However, it suffices to take this upper bound over the *acyclic paths* in $F$; an acyclic path is a path $p$ which is not Nerode equivalent to any of its proper prefixes. If we are rewriting a finite feature structure $F$, then there are a finite number of such paths. The result of the parallel rewriting is another finite feature structure. Each parallel rewriting step can then be simulated by a finite number of sequential steps.

□

We have completed the preliminary results of the section; now we can go on to the characterization of the resolution $\overline{F}$ of a feature structure using default extensions.

**Definition 5.4** *Let $S$ be a set of constraints. Define $\Delta(S)$, the default set determined by $S$, to be the set*

$$\{(pC_1, pC_2) \mid p \in L^*, (C_1, C_2) \in S\}.$$

We start with a lemma about extensions in general.

**Lemma 5.6** *Let $D$ be a Scott domain and $\Delta$ a default set. If $y$ is an extension of $x$ and $(a, b)$ is a default such that $a \sqsubseteq y$ and $b \uparrow y$ then $b \sqsubseteq y$.*

**Proof.** This is straightforward, but it gives a typical use of techniques for reasoning with extensions, so we give the details. By definition of extensions (Def. 2.3) we have that $\phi(x, y) = \bigsqcup_i \phi(x, y, i) = y$. If $a \sqsubseteq y$ then since $a$ is compact, $a \sqsubseteq \phi(x, y, i)$ for some $i$. Thus $\phi(x, y, i) \sqcup b \sqsubseteq \phi(x, y, i + 1)$, whence $b \sqsubseteq \phi(x, y) = y$.

□

**Theorem 5.2** *Fix a finite set $S$ of constraints. Let $F$ be any feature structure such that $\overline{F}$ exists with respect to $S$. Then $\overline{F}$ is the unique default extension of $F$ with respect to $\Delta(S)$.*

**Proof.** Suppose that $F$ is such that $\overline{F}$ exists. We show that every extension of $F$ subsumes $\overline{F}$. This proves that there is a unique extension, because by Theorem 2.1, distinct extensions must be inconsistent, and we know that extensions always exist. Let $E$ be an extension of $F$. We prove by induction that $\phi(F, E, i) \sqsubseteq \overline{F}$. The basis is clear. Assume that $\phi(F, E, i) \sqsubseteq \overline{F}$. Now

$$\phi(F, E, i+1) = \phi(F, E, i) \sqcup \bigsqcup\{pC_2 \mid (pC_1, pC_2) \in \Delta \ \& \ pC_1 \sqsubseteq \phi(F, E, i) \ \& \ pC_2 \uparrow E\}.$$

But since $\phi(F, E, i) \sqsubseteq \overline{F}$ we have that if $pC_2$ occurs on the right side of the above equation, then $pC_2 \sqsubseteq \overline{F}$ because $\overline{F}$ is resolved. Thus $\phi(F, E, i + 1) \sqsubseteq \overline{F}$, and therefore $E = \phi(F, E) \sqsubseteq \overline{F}$.

We next show that $E$ must be resolved. This is because $E \sqsubseteq \overline{F}$. Suppose that $pC_1 \sqsubseteq E$; then the corresponding $pC_2 \sqsubseteq \overline{F}$. Since $E \sqsubseteq \overline{F}$, we have that $pc_2 \uparrow E$. Therefore $pC_2 \sqsubseteq E$, by Lemma 5.6. This completes the proof, since now $E = \overline{F}$.

□

The converse of this theorem is false. Let $F$ be the structure $[f : a]$ and consider the constraint set $S = \{([f : \bot], [f : b])\}$, where $a$ is inconsistent with $b$. Then $F$ has itself as its own unique $\Delta(S)$-extension, but there is no corresponding $\overline{F}$. The best we can do at present is the following.

**Theorem 5.3** *Let $F$ have an extension $E$ with respect to $\Delta(S)$ and suppose that all $(pC_1, pC_2)$ in $\Delta(S)$ such that $pC_1 \sqsubseteq E$ are such that $pC_2 \uparrow E$. Then $E = \overline{F}$.*

**Proof.** Such an $E$ must be resolved, because if there is a default $(pC_1, pC_2)$ with $pC_1 \sqsubseteq E$, then $pC_2 \uparrow E$ by hypothesis. By Lemma 5.6, we have that $pC_2 \sqsubseteq E$. This proves that $\overline{F}$ exists, and so $\overline{F} \sqsubseteq E$. But from the proof of the previous theorem, $E \sqsubseteq \overline{F}$.

□

## 6. Disjunctive constraints

Now we turn to a much more difficult problem: the case when constraints have the form $(\phi, \psi)$ with $\psi$ a disjunction of conjunctive formulas. As an exampler, consider the FCR

$$([\mathrm{PRD}] \;\&\; [\mathrm{VFORM}]) \supset ([\mathrm{PAS}] \vee [\mathrm{PRP}])$$

from GPSG.

To formalize this sort of thing, we begin again with Carpenter's definition. This time, though, constraints have the form $C = (C_1, \mathbf{C_2})$ where $\mathbf{C_2}$ is a *set* of feature structures: the set of most general satisfiers of the formula $\psi$.

**Definition 6.1** *Let $F$ be a feature structure, and $C$ a (single) disjunctive constraint of the above form. We say that $F$ is* resolved *with respect to $C$ iff for any path $p$ of $F$, and any constraint $(C_1, \mathbf{C_2})$ in $S$,*

$$C_1 \sqsubseteq F/p \Rightarrow (\exists C_2 \in \mathbf{C_2})(C_2 \sqsubseteq F/p).$$

*If $S$ is a set of constraints, we say that $F$ is resolved with respect to $S$ if it is resolved with respect to $C$ for any $C \in S$.*

Our objective is an analogue of Theorem 5.2, characterizing (at least some of) the resolved feature structures subsumed by a given feature structure. Now, though, there may be no least such structure. Consider the constraint set

$$\{(f : \bot, \{f : a, f : b\})\}$$

where $a, b$ and $\bot$ are sorts. There are two minimal resolved structures extending $f : \bot$; namely $f : a$ and $f : b$. In some sense the set of these two structures is the "minimum" constrained object extending the given one, which should now be itself considered as a singleton set. Our problem is to capture this idea using a reasonable domain, and then to define defaults in this domain so as to obtain our unique extension theorem. Before doing this, though, we remark that the form $(C_1, \mathbf{C_2})$ is the most general we will consider, because if the first element $C_1$ of the pair were a set, then we could replace the set of pairs $(\mathbf{C_1}, \mathbf{C_2})$ by the set of pairs

$$\{(C_1, \mathbf{C_2}) \mid C_1 \in \mathbf{C_1}\}.$$

The payoff for all of this work may not be obvious; but it lies precisely in getting a good idea for what kind of defaults work well when we want to add disjunctive default information to an element of a domain, or to a disjunction of such elements. One might expect that a standard version of default feature logic would do the job. This does not work, though. It turns out that we run into the "or problem" considered by workers in default logic (see, e.g., Poole [19], or Gelfond *et al.* [6].) For example, consider the default rules

$$\frac{p : q}{q} \text{ and } \frac{r : q}{q}.$$

If we start with the formula $p \vee r$ then neither precondition of the rules can be established (by proof), so neither is applicable; but it seems that we should be able to add the information $q$ (conjunctively) to our stock. We have the same problem in the setting of feature structures; and it re-occurs in a domain-theoretic generalization. For the generalization, and in fact to handle the specific case of feature structures, we use the *Smyth powerdomain.* We begin with a review of this construction.

### *6.1 The Smyth powerdomain*

Each feature structure represents a conjunctive piece of information: every new path tells us a bit more about the object represented by the structure. We now want a domain construction such that an element of the constructed domain represents a *disjunctive* possibility, as in the case of **KR** logic. One might think that the generalization of two feature structures represented the disjunction of the information in them, because unification represents conjunction. But this is easily seen to be incorrect, as generalization does not distribute over unification.

When we have disjunction in our logic, we note that the set of all minimal satisfiers of a formula is a finite *antichain* of pairwise-incomparable compact elements; these structures do not subsume each other. We take these collections as representatives of (finite) disjunctive structures. Then the question becomes: how to order such antichains? From the logical perspective, an antichain $\{F_1, \ldots F_n\}$ represents an object which is more specific than at least one of the $F_i$. We can thus increase the informativeness of the antichain by removing one or more of the elements, or by increasing the informativeness of one or more of the elements. This applies to general Scott domains.

**Definition 6.2 (Smyth (upper) subsumption)** *Let $X$ and $Y$ be finite antichains on a domain $(D, \sqsubseteq)$. We say $X \sqsubseteq^\sharp Y$ iff for all $e \in Y$, there is some $d \in X$ with $d \sqsubseteq e$.*

It is easy to see that $\sqsubseteq^\sharp$ is a partial order on antichains.

This construction actually does most of the work for us when we want to calculate with disjunctive sets. For example, to unify (conjoin) $X$ and $Y$ we form the set

$$\{x \sqcup y \mid x \in X; y \in y\}$$

and remove non-minimal elements. The formula is expected given that the distributive law holds in the logic of the domain of feature structures. But in fact the construction is general. Furthermore, to "disjoin" $X$ with $Y$ we form $X \cup Y$ and remove nonminimal elements.

In effect we have now constructed the set of compact elements of the Smyth powerdomain of a Scott domain. To generate the "complete" complete partial order "determined" by these compact elements, we use a domain-theoretic technique known as *ideal completion.* This is a general technique which always produces an algebraic cpo from a partially ordered set of elements, in such a way that all existing joins (but not meets) are preserved. The details in general are as follows.

**Definition 6.3 (Ideal Completion)** *Given a partially ordered set $(K, \preceq)$, we define an ideal of $K$ to be a nonempty, directed, and downward-closed subset $I$ of $K$. (The last condition just says that if $X \in K$ and $U \preceq X$ then $U \in K$.) Then the ideal completion of $(K, \preceq)$ is the collection of ideals of $K$, partially ordered by inclusion of sets.*

We take $K$ to be the set of *nonempty*[2] finite antichains of compact elements of $D$, and we take $\preceq$ to be $\sqsubseteq^\sharp$. and arrive at the Smyth or *upper* powerdomain the set of ideals of $K$, ordered by set inclusion. It is possible to prove that if one starts with a Scott domain, then the Smyth powerdomain is again a Scott domain. Moreover, and this is easy to see, the mapping sending an antichain $X$ to its "downward closure" $\downarrow X = \{Y : Y \sqsubseteq^\sharp X\}$ is an injection preserving the order $\sqsubseteq^\sharp$.

One can use other orderings on sets of domain elements to get new powerdomains. Two other common constructions, for example, are the *Hoare* (lower) and the *Plotkin* (convex) powerdomains. See Gunter and Scott [8] for a survey. More powerdomains can be found in the work of Buneman *et al.* [2], Gunter [7], and Libkin [14]. In fact we use Libkin's ideas on Smyth antichains as "or-sets" to get a proper notion of defaults in the present setting.

### *6.2 Updates and resolution*

We know, from the beginning of the section, what it means for a feature structure to be resolved with respect to a disjunctive constraint $(C_1, \mathbf{C_2})$. We extend this definition to antichains by simply saying

[2]We could include the empty antichain, but this would add an "inconsistent" element to our powerdomain, which we will in general not want.

that an antichain $X = \{F_1, \ldots, F_n\}$ is resolved whenever each $F_i$ is resolved.

We are going to consider a crucial definition, due to Libkin [14]. We will state it for general domains; for this we need to generalize disjunctive constraints.

**Definition 6.4** *A* disjunctive constraint *over a domain $D$ is a pair $(a, B)$, where $a \in \kappa(D)$ and $B$ is a nonempty finite antichain over $D$.*

The idea of this is that if a finite piece of information $d$ entails the precondition $a$ of the constraint, then we can "improve" $d$ by adding one of the information pieces in $B$. This is captured, though, by using antichains of which $d$ is a member.

**Definition 6.5 (Update (Libkin)).** *For an antichain $X$ over $\kappa(D)$, and for $d \in X$ and an antichain $B$ over $\kappa(D)$, let the* update *of $X$, written*

$$X[d \leftarrow d \sqcup B],$$

*be the set*

$$min((X \setminus \{d\}) \cup \{d \sqcup b \mid b \in B\}),$$

*where $min(Z)$ removes nonminimal elements from a set $Z$. If the result of the whole operation is empty, we declare the operation undefined.*

Notice that the update $X[d \leftarrow d \sqcup B]$ improves the element $d$, leaving the rest of the elements of $X$ unchanged. (It could actually remove the element $d$ if that element were inconsistent with each element of $B$.) So $X \sqsubseteq^\sharp X[d \leftarrow d \sqcup B]$.

We want to apply these ideas using Definition 6.1 and its generalization to antichains. Those definitions refer to arbitrary paths $p$ occurring in feature structures. In effect, instead of a constraint $(C_1, \mathbf{C_2})$, we are working with a set of constraints, all of the form

$$(pC_1, \{pC_2 \mid C_2 \in \mathbf{C_2}\})$$

in the sense of Definition 6.4, over the domain of feature structures. So in what follows, we will just drop the notation "$p$" from our constraints; this will increase readability.

**Lemma 6.1** *Let $X$ be a finite antichain over the domain of feature structures, and let $C = (C_1, \mathbf{C_2})$ be a constraint. Then $X$ is resolved with respect to $C$ iff for each $F \in X$, if $C_1 \sqsubseteq F$ then*

$$X[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp X.$$

**Proof.** Suppose that $X$ is resolved with respect to $C$. Let $F \in X$ and $C_1 \sqsubseteq F$. Since $X$ is resolved, there is a $C_2 \in \mathbf{C_2}$ with $F \sqcup C_2 = F$. Looking at the definition of update, we see that the operation of throwing out non-minimal elements (to get an antichain) will replace all elements in $F \sqcup \mathbf{C_2}$ by $F$. So

$$X[F \leftarrow F \sqcup \mathbf{C_2}] = X.$$

Conversely, suppose the update condition

$$X[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp X.$$

for every $F \in X$. We want to show that $X$ is resolved. By the definition of $\sqsubseteq^\sharp$, if I pick $F$ in $X$ on the right, then $F$ will be subsumed (in the domain of feature structures) by some element of the update on the left side. This element cannot be in $X \setminus \{F\}$, for this would violate the antichain condition. Thus for some $C_2$, $F \sqcup C_2 \sqsubseteq F$.

□

**Remark.** When we use a relationship like

$$X[F \leftarrow F \sqcup \mathbf{C_2}] = X$$

we are implicitly assuming that the given update is defined. This remark applies to the subsequent uses of the notation later on.

Next we look at resolution in infinite Smyth elements. The form of our definition is suggested by Lemma 6.1.

**Definition 6.6** *Let $X$ be an element of the Smyth powerdomain of the domain of feature structures. We say that $X$ is resolved with respect to a constraint $C = (C_1, \mathbf{C_2})$ iff for any finite antichain $A$ with $A \sqsubseteq^\sharp X$ (that is, as an ideal $\downarrow A \subseteq X$), and for any $F \in A$:*

$$C_1 \sqsubseteq F \Rightarrow A[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp X.$$

We justify this definition by showing that it reduces to our earlier one in case $X$ is in fact finite. Say that $X$ is $\infty$-resolved (with respect to $C$) if it satisfies the definition just given.

**Lemma 6.2** *Fix $C$. If $X$ is a finite antichain, then $X$ is resolved iff $X$ is $\infty$-resolved with respect to $C$.*

**Proof.** One direction of this is trivial; we prove only the more difficult direction. Suppose that $X$ is resolved. Let $A$ be a finite antichain with $A \sqsubseteq^\sharp X$ (as antichains.) Pick $F \in A$ with $C_1 \sqsubseteq F$. Now in $X$ do the following. Find all $G \in X$ such that $F \sqsubseteq G$; call this set $\{G_1, \ldots, G_n\}$. (It may be empty.) Form the "iterated update"

$$I = X[G_1 \leftarrow G_1 \sqcup \mathbf{C_2}] \ldots [G_n \leftarrow G_n \sqcup \mathbf{C_2}].$$

It is easy to check that this does not depend on the order of listing $\{G_1, \ldots, G_n\}$. Further, since $X$ is resolved, by induction the iterated update $I$ is actually equal to $X$. On the other hand, by construction,

$$A[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp I.$$

This shows that $X$ is $\infty$-resolved with respect to $C$.

□

*6.3 Minimal resolved elements and default extensions*

We now have most of the technical machinery to extend our earlier result (Theorem 5.2) to the disjunctive case. Recall that the idea in that case was to find the most general resolved feature structure $\overline{F}$ extending $F$, assuming that there was one. We start with a feature structure $F$ again, but now consider it as a finite antichain $\{F\}$ in the Smyth powerdomain of feature structures. (We could in fact start with any finite antichain.) Now the idea is to find the "minimum" Smyth "set" which is resolved and extends $\{F\}$ in the Smyth ordering. (By "resolved" here we mean with respect to all of the given constraints.)

**Definition 6.7** *Assume that $\{F\} \sqsubseteq^\sharp X$ for some resolved Smyth element $X$. By $\overline{F}$ we mean the Smyth generalization*

$$\bigsqcap \{X \mid X \text{ is resolved and } \{F\} \sqsubseteq^\sharp X\}.$$

One can think of $\overline{F}$ as the set of all minimal finite or infinite feature structures which are resolved and which $F$ subsumes. To see why, assume for a moment that $\overline{F}$ is a finite Smyth antichain $\{G_1, \ldots, G_n\}$. We will show below that this antichain is resolved, so all the feature structures $G_i$ must be resolved. But they must also be minimal resolved structures, because replacing any of them by a strictly less informative resolved structure would result in an antichain which was resolved and strictly lower in the Smyth order than $\{G_1, \ldots, G_n\}$. Assume that, for example, we replace $G_1$ by a strictly smaller resolved $G'_1$ (still subsumed by $F$, of course). Call the new antichain $G' = \{G'_1, G_2, \ldots, G_n\}$. Then we have that $G' \sqsubset^\sharp \overline{F}$. On the other hand, $G'$ is resolved, so $\overline{F} \sqsubseteq G'$, a contradiction.

**Lemma 6.3** *If $\overline{F}$ exists, then it is resolved.*

**Proof.** We work with the ideal completion. The Smyth generalization $\overline{F}$ of the set of all resolved ideals $X$ with $\downarrow \{F\} \subseteq X$ is just the infinite intersection of all such ideals $X$. (This is where we need the existence of one such ideal.) Now this intersection must be resolved, for let a constraint $C = (C_1, \mathbf{C_2})$ be given and let $A$ be a finite antichain with $\downarrow A \subseteq \overline{F}$, and $C_1 \sqsubseteq F$ for some $F \in A$. Then since $\downarrow A \subseteq X$ for each of the resolved ideals $X$, we have

$$A[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp X.$$

Therefore the updated antichain, as an ideal, is a subset of $\overline{F}$, and so $\overline{F}$ is resolved with respect to $C$.
□

We are now completely ready for the default definitions. The form of updates tells us what to use for defaults in the Smyth powerdomain.

**Definition 6.8** *Let $S$ be a set of constraints, each of the form $C = (C_1, \mathbf{C_2})$. The* Smyth default set *determined by $S$ is*

$$\Delta(S) = \{(A, A[F \leftarrow F \sqcup \mathbf{C_2}]) \mid \textit{ A finite antichain, } (C_1, \mathbf{C_2}) \in S,\ F \in A,\ C_1 \sqsubseteq F\}.$$

That is to say that the default set consists of all pairs of antichains such that the second pair element can in some way be obtained from the first by updating according to a constraint in $S$.

Having the set of defaults to use, we can apply the standard definition of extension, valid in any domain, to the Smyth powerdomain. We repeat the definition here using capital letters for domain elements: $\phi(X, Y) = \bigsqcup_i \phi(X, Y, i)$, where $\phi(X, Y, 0) = X$, and

$$\phi(X, Y, i+1) = \phi(X, Y, i) \sqcup \bigsqcup \{B \mid (A, B) \in \Delta \ \&\ A \sqsubseteq \phi(X, Y, i) \ \&\ B \uparrow Y\}$$

for all $i \geq 0$. We say that $E$ is an extension of $X$ if $\phi(X, E) = E$.

**Theorem 6.1** *Fix a set of constraints $S$. Let $F_0$ be a feature structure, and assume that $\overline{F_0}$ exists in the Smyth powerdomain. Then $\overline{F_0}$ is the unique default extension of $\{F_0\}$ with respect to the default set $\Delta(S)$.*

**Proof.** We have only to emulate the proof of Theorem 5.2 using our new default set. So we show by induction that for any extension $E$ of $\{F_0\}$,

$$\phi(\{F_0\}, E, i) \sqsubseteq^\sharp \overline{F_0}.$$

The basis is clear. Assume the result for $i$. Then $\phi(\{F_0\}, E, i+1)$ is obtained by taking $\phi(\{F_0\}, E, i)$ and adding elements of the form

$$A[F \leftarrow F \sqcup \mathbf{C_2}]$$

where $A \sqsubseteq^\sharp \phi(\{F_0\}, E, i)$, $F \in A$, $C_1 \sqsubseteq F$, $(C_1, \mathbf{C_2}) \in S$, and $A[F \leftarrow F \sqcup \mathbf{C_2}] \uparrow E$. But $\phi(\{F_0\}, E, i) \sqsubseteq^\sharp \overline{F_0}$ and $\overline{F_0}$ is resolved. So any of the added updates subsume $\overline{F_0}$, which implies

$$\phi(\{F_0\}, E, i+1) \sqsubseteq^\sharp \overline{F_0}.$$

This proves, as in the case of Theorem 5.2, that there is a unique extension $E$ of $\{F_0\}$. We complete the proof by showing that $E$ is resolved. We already know $E \sqsubseteq \overline{F_0}$. Apply the definition of resolved Smyth element. Take an antichain $A$ and $F \in A$ with $C_1 \sqsubseteq F$ for a constraint $(C_1, \mathbf{C_2})$ in $S$. Assume $A \sqsubseteq^\sharp E$. Then $A \sqsubseteq^\sharp \overline{F_0}$. But $\overline{F_0}$ is resolved, so

$$A[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp \overline{F_0}.$$

Since $E \sqsubseteq^\sharp \overline{F_0}$, we have that

$$A[F \leftarrow F \sqcup \mathbf{C_2}] \uparrow E.$$

Then $A[F \leftarrow F \sqcup \mathbf{C_2}] \sqsubseteq^\sharp E$ by Lemma 5.6.

□

**Example.** The theorem above does not provide rules for calculating extensions, but in most cases this is not too difficult, if we are willing to make some simplifications. Consider the constraint set of our earlier example

$$\{([f:\bot],\{[f:a],[f:b]\})\}$$

where $a, b$ and $\bot$ are sorts. There seem to be two minimal resolved structures extending $[f:\bot]$; namely $[f:a]$ and $[f:b]$. We show that $\{[f:a],[f:b]\}$ is the minimum antichain which completes the set $\{[f:\bot]\}$. In fact, any element $G$ in another resolved antichain $A$ extending $\{[f:\bot]\}$ must be subsumed by $[f:\bot]$, and so must also be subsumed by either $[f:a]$ or $[f:b]$. That means that the antichain $\{[f:a],[f:b]\} \sqsubseteq^\sharp A$.

What happens when we calculate extensions? Suppose our feature signature allows another feature name $g$. What antichains actually subsume the singleton set $\{[f:\bot]\}$? We could use the antichain

$$A = \{[g:\bot],[f:\bot]\}$$

obtaining as part of the extension of $\{[f:\bot]\}$ at the first iteration the antichain

$$\{[g:\bot, f:\bot],[f:a],[f:b]\}.$$

In fact we could use any antichain of feature structures consistent with $[f:\bot]$ as part of the first approximation to the extension, so that the "most general" disjunctive Smyth element extending $\{f:\bot\}$ might be thought of as having the form

$$\overline{\{f:\bot\}} = \{G \sqcup [f:\bot] : G \uparrow [f:\bot]\} \cup \{[f:a],[f:b]\}$$

where the $G$ are restricted to occur in an antichain. But the problem is not so difficult. Assuming that $a$ and $b$ are the only sorts, which can label any node of a feature structure, and that $f$ and $g$ are the only feature names, then the first approximation

$$\phi(\{[f:\bot]\},\{[f:a],[f:b]\},1) = \{a[f:\bot], b[f:\bot],[g:\bot,f:\bot],[f:a],[f:b]\}.$$

This corresponds to the fact that the elements $a$, $b$, and $[g:\bot]$ are minimal feature structures incomparable with $[f:\bot]$.

At this point it looks like "irrelevant" information is entering our extension. However, this is a feature of Smyth-style approximation, as pointed out by Pollard and Moshier [17]. Notice that the above antichain is not resolved. For example, the structure $a[f:\bot]$ will be replaced by $\{a[f:a], a[f:b]\}$ in the next iteration. These two elements will be subsumed by the already existing elements $[f:a]$ and $[f:b]$, so at the next stage, we will have

$$\phi(\{[f:\bot]\},\{[f:a],[f:b]\},2) = \{[f:a],[f:b]\}.$$

In general, Smyth approximation starts with the most disjunctive possibilities, and "cuts down" at successive levels. So normally, we do not have to worry about extending with irrelevant information.

On the other hand, we must include "irrelevant" information as part of our default sets. Suppose we had another default constraint of the form $([f:a],\{[g:b]\})$. We cannot just use the singleton default set

$$\{(\{[f:a]\},\{[g:b]\})\},$$

because then we could not apply the default to the structure formed at the first iteration; *every* element of that antichain would have to be subsumed by $[f:a]$. Singleton antichains are very informative elements; it is hard to make them subsume anything in the $\sqsubseteq^\sharp$ ordering. Here we see the solution, therefore, to the "or-problem" raised earlier; it lies in adding "irrelevant" information to antichains used in defaults.

**A second example.** Consider the domain for birds mentioned in Section 2.1. Assume just one constraint of the form $(bird, \{fly\})$, which is nondisjunctive. How does this example fare with our update semantics? Let us compute Smyth extensions of $\{tweet\}$. At the first stage, the most general antichain subsuming $\{tweet\}$ is $\{peng, tweet, fly\}$. There are two updates of this: $\{peng, fly\}$ and $\{tweet, fly\}$ (the second comes because we have "knocked out" penguins by updating them with the inconsistent flying type). Unifying these two updates with $\{tweet\}$ gives the antichain

$$X = \{pengtweet, ftweet\}.$$

This is the first approximation to an extension. Is it in fact an extension? The atom *pengtweet* is not resolved, so according to our theory we have not arrived at an extension. And in fact $X$ itself is an updatable antichain, because *pengtweet* is subsumed by *bird*; the update $X[pengtweet \leftarrow pengtweet \sqcup fly]$ is just $\{ftweet\}$, because we have again knocked out penguins. Thus the unique extension of $\{tweet\}$ is $\{ftweet\}$.

### *6.4 Arbitrary disjunctive constraints*

In this short section we remark that our constructions apply to general Scott domains. Recall the definition of general disjunctive constraint: Def. 6.4. Resolution is a well-defined concept for general domain elements (Def. 6.1). Updates are valid constructions in any domain; Lemma 6.1 applies to general domains, as does lemma 6.2. All of the results of the section, in fact, carry over in general. So we have a general method for augmenting a (disjunctive) element of the Smyth powerdomain with disjunctive default information. Applications of these observations, though, are beyond the scope of the paper.

## 7. Non-monotonic Feature Logic

We now have a model theory for disjunctive default information. In this section we show how to use this model theory to get a notion of *non-monotonic consequence* between formulas of **KR** logic.

First of all, notice that constraints may admit no resolved structures. For example, consider the set of two constraints

$$S = \{(\perp, \{a, b\}), (\perp, \{c, d\})\}$$

where $a$ and $b$ are each inconsistent wth $c$ and $d$. Then $\{a, b\}$ and $\{c, d\}$ are both extensions of $\{\perp\}$, and there is no minimal resolved extension.

Many authors, starting with Reiter, have considered how to define the notion of "default theorem". Here we will be content with a semantic notion of entailment between formulas. The basic idea is that for a formula $\beta$ to be entailed by a formula $\alpha$, the information in $\beta$ should be common to all the extensions of the formula $\alpha$. Of course this is not very precise; but we have the tools at hand to make a precise definition.

We start by fixing a set of constraints $S$, and then forming $\Delta(S)$, the Smyth default set determined by $S$. Let $\alpha$ and $\beta$ be satisfiable Kasper-Rounds formulae. By $Minsat(\phi)$ we denote the antichain of minimal satisfiers of $\phi$.

**Definition 7.1** *We say that $\beta$ is a nonmonotonic consequence of $\alpha$ with respect to $\Delta(S)$, written $\alpha \mid\!\sim_{\Delta(S)} \beta$, if and only if*

$$Minsat(\beta) \sqsubseteq^\sharp X$$

*for any extension $X$ of $Minsat(\alpha)$ with respect to $\Delta(S)$.*

Notice here that we interpret formulae by their set of minimal satisfiers; so instead of taking extensions of singleton antichains, we are taking extensions of compact Smyth elements in general.

As an example, consider the constraints given just above. Let $\alpha$ be the fomula **true**, and let $\beta$ be $a \vee b \vee c \vee d$. Then $\{a, b, c, d\} \sqsubseteq^\sharp \{a, b\}$ and also $\{c, d\}$, so that $\beta$ is a non-monotonic consequence of $\alpha$. Since $\{a, b, c, d\}$ is the generalization of the two extensions, $a \vee b \vee c \vee d$ is the strongest nonmonotonic consequence of **true** up to logical equivalence.

We next consider the basic Kraus-Lehmann-Magidor [12] axioms for "reasonable" notions of preferential entailment. A "core" set of laws is the following:

1. (Reflexivity) $\varphi \mid\!\sim \varphi$.
2. (Left Logical Equivalence) If $\models (\varphi \leftrightarrow \psi)$, and $\varphi \mid\!\sim \beta$, then $\psi \mid\!\sim \beta$.
3. (Right Weakening) If $\models (\alpha \rightarrow \beta)$ and $\varphi \mid\!\sim \alpha$, then $\varphi \mid\!\sim \beta$.
4. (Cut) If $\varphi \wedge \alpha \mid\!\sim \beta$, and $\varphi \mid\!\sim \alpha$, then $\varphi \mid\!\sim \beta$.
5. (Cautious monotony) If $\varphi \mid\!\sim \alpha$ and $\varphi \mid\!\sim \beta$, then $\varphi \wedge \alpha \mid\!\sim \beta$.
6. (Or) If $\alpha \mid\!\sim \gamma$ and $\beta \mid\!\sim \gamma$, then $\alpha \vee \beta \mid\!\sim \gamma$.

How does our definition measure up to these laws? It follows by general properties of domain-theoretic extensions that the first four laws hold. See [27] for discussion. In general the Cautious Monotony law does not hold. For completeness we adapt an example from [27].

**Example.** Consider the Scott domain consisting of four sorts $\{\bot, a, b, b'\}$. These are ordered by $\bot \sqsubseteq a$; $a \sqsubseteq b$, and $a \sqsubseteq b'$, where $b$ is inconsistent with $b'$. Consider the following two default constraints, expressed without being embedded as antichains:

$$\{(\bot, \{b\}), (a, \{b'\})\}.$$

We claim that with these constraints, we have that **true** $\mid\!\sim b$ and **true** $\mid\!\sim a$, but not (**true** $\wedge\, a) \mid\!\sim b$. To verify this, check that the unique Smyth extension of $\{\bot\}$ is $\{b\}$, whence **true** $\mid\!\sim b$; it follows that **true** $\mid\!\sim a$. However, the Smyth element $\{a\}$ has two extensions $\{b\}$ and $\{b'\}$. So (**true** $\wedge\, a) \equiv a$ does not nonmonotonically entail $b$.

$\square$

Cautious monotony does hold if extensions are unique, and in some other cases (see the same reference). Finally, we do not know the status of the Or law. We think that it should hold, but we have no proof at the moment.

## 8. Logics and Inheritance

In this section we return to more specific logics for feature structures. We consider Dawar and Vijayshanker's three-valued logic [4], which allows negation; and we combine this with Carpenter's appropriateness specifications. Doing this also requres a change in the definition of feature structures, as we must now include negative information in them. This we do using Carpenter's notion of (weakly) inequated feature structures. Our only concern is that the new domain of feature structures remains a BCPO and that formulas in the new logic still have a finite number of minimal (positive) satisfiers. It follows then that the Smyth techniques above still apply.

In the last part of the section, we sketch briefly a method for incorporating defaults into an inheritance hierarchy. It involves a simple notion of "layering" extensions, given that default constraints are associated with each sort in the hierarchy.

### *8.1 Feature logic with negation*

First we review Carpenter's appropriateness conditions. We henceforth fix a finite set of sorts $A$, and assume that $A$ is a BCPO under an ordering $\sqsubseteq_A$, where we drop the subscript when there is no ambiguity. We also assume that the set of feature names $L$ is finite.

**Definition 8.1 (Carpenter appropriateness)** *An* appropriateness specification *over the poset* $(A, \sqsubseteq)$ *is a partial function* $Approp : L \times A \rightarrow A$ *such that*

- *For every feature name* $f \in L$*, there is a minimum (most general) sort* $Intro(f)$ *such that* $Approp(f, Intro(f))$ *is defined;*
- *If* $Approp(f, a)$ *is defined and* $a \sqsubseteq b$ *then* $Approp(f, b)$ *is also defined and*

$$Approp(f, a) \sqsubseteq Approp(f, b).$$

If the pair $(f, a)$ is in the domain of *Approp* we say that feature $f$ is appropriate for the sort $a$. If $Approp(f, a) = c$ then we say that in addition $c$ is an appropriate sort of value for the feature $f$. We will have little to say about appropriate values, so in what follows we will make the simplifying assumption that $Approp(f, a) = \bot$ whenever $Approp(f, a)$ is defined. Then the conditions simply say that for each feature $f$, the set of sorts for which $f$ is appropriate is a principal filter over $A$. If $Approp(f, a)$ is, on the other hand, *undefined*, we regard this as definite information about the sort $a$: if a path of a feature structure has sort $a$, then it is not possible to give the feature $f$ a value at that point. We formalize this in the following definition.

**Definition 8.2** *An abstract feature structure* $F$ *is said to be* well-typed *if whenever* $f \in L$*, and* $p$ *and* $pf$ *are paths of* $F$*, then* $f$ *is appropriate for* $\theta(p)$.

Carpenter [3, Theorem 52] shows that the collection of well-typed feature structures is a sub-BCPO of the collection of feature structures; the ordering $\sqsubseteq_A$ restricted to the well-typed structures still is a Scott domain.

Next, in preparation for introducing negation into the logic, we review the notion of an inequated structure.

**Definition 8.3** *Let* $F = (T, N, \theta)$ *be an abstract feature structure. An* inequation relation *is a relation* $Z$ *on* $T$ *which is symmetric and such that* $Z \cap N = \emptyset$.

The relation $Z$ is intended to indicate that pairs of paths standing in the relation can never be made equal. In order to make this idea precise, we augment feature structures with such relations.

**Definition 8.4** *An abstract inequated feature structure is a tuple* $F = (T, N, Z, \theta)$ *where* $Z$ *is an inequation relation on* $N$.

The subsumption order is extended to inequated structures by requiring the $T, N$, and $\theta$ components to be ordered by inclusion as usual, an by additionally requiring inclusion between the $Z$ components. Once again Carpenter shows that inequated structures form a Scott domain [3, Theorem 80]. It is straightforward to show as well that the collection of well-typed and inequated structures is again such a domain. By "feature structure" we now mean an abstract, well-typed, and negated structure.

We are almost ready for our new logic with negation. One more preliminary definition paves the way. Let $p$ be an arbitrary string of labels, and let $F$ be a feature structure. We say that $F$ is *strongly inappropriate* for $p$ iff there is some prefix $r$ of $p$ and label $f$ such that $r$ is a path of $F$, $rf$ is a prefix of $p$, and $f$ is inappropriate for any type $b \sqsupseteq \theta(r)$. The point of this definition is that when we consider a path equation $p \doteq q$, we will know that such a path equation could never be satisfied by a feature structure $F$ or any of its more specific extensions if $F$ were strongly inappropriate for either $p$ or $q$. Similarly, when we consider a formula $f : \phi$, we will know that a structure $F$ or any of its extensions could not satisfy this if $F$ were strongly inappropriate for $f$.

**Definition 8.5 (DV logic).** *The syntax of Dawar-Vijayshanker logic is obtained from KR logic by adding the unary logical connective* $\neg$.

The interesting part of the logic, as pointed out by Dawar and Vijayshanker, is its Kleene three-valued semantics.

**Definition 8.6** *We define two relations $\models^+$ and $\models^-$ inductively on the structure of* **DV** *formulas.*

- $F \models^+$ **true** *always;*
- $F \models^-$ **true** *never;*
- *Similarly for* **false**, *reversing $\models^+$ and $\models^-$;*
- $F \models^+ a$ *iff* $a \sqsubseteq \theta(\epsilon)$;
- $F \models^- a$ *iff* $a$ *is inconsistent with* $\theta(\epsilon)$;
- $F \models^+ (p \doteq q)$ *iff* $(p, q) \in N$;
- $F \models^- (p \doteq q)$ *iff $F$ is strongly inappropriate for either $p$ or $q$, or $p$ and $q$ are paths of $F$ and* $(p, q) \in Z$;
- $F \models^+ \neg\phi$ *iff* $F \models^- \phi$;
- $F \models^- \neg\phi$ *iff* $F \models^+ \phi$;
- $F \models^+ f : \phi$ *iff* $F/f \models^+ \phi$;
- $F \models^- f : \phi$ *iff $F$ is strongly inappropriate for $f$ or* $F/f \models^- \phi$;
- $F \models^+ (\phi \vee \psi)$ *iff* $F \models^+ \phi$ *or* $F \models^+ \psi$;
- $F \models^- (\phi \vee \psi)$ *iff* $F \models^- \phi$ *and* $F \models^- \psi$;
- $F \models^+ (\phi \wedge \psi)$ *iff* $F \models^+ \phi$ *and* $F \models^+ \psi$;
- $F \models^- (\phi \wedge \psi)$ *iff* $F \models^- \phi$ *or* $F \models^- \psi$.

What can we say about this logic? Dawar and Vijayshanker prove, for example, that it has the *persistence property*; if $F \sqsubseteq G$ and $F \models^+ \phi$ then $G \models^+ \phi$. Their proof does not consider an explicit notion of negated feature structures, but is easily extended to this case. Also, minor modifications are needed because their sort hierarchy is a flat ordering; this is accommodated by our definition of strong inappropriateness.

More to the point for us is the following result:

**Proposition 8.1** *Any formula in* **DV** *has a finite number of minimal $\models^+$ satisfiers and a finite number of minimal $\models^-$ satisfiers.*

**Proof.** One can prove this directly by a simultaneous induction on the structure of formulas. The base cases are straightforward, with two exceptions. First, the minimal $\models^-$-satisfiers of the atomic formula $a$ are

1. all those one-node feature structures having a minimal sort inconsistent with $a$; together with
2. all structures of the form $b[f : \bot]$, where $b$ is a minimal sort such that (i) $b$ is appropriate for $f$; (ii)$a$ (as a feature structure) is strongly inappropriate for $f$; and (iii) $b \uparrow a$.

Finiteness of both $A$ and $L$ is needed here.

Likewise, the minimal $\models^-$-satisfiers of $p \doteq q$ are

1. single-path structures $u$ which are proper prefixes of $p$, typed at each point with a minimal sort appropriate for the next feature name, except at the last node, where they have a minimal sort $b$ such that than $b$ is strongly inappropriate for the feature $f$, where $uf$ is a prefix of $p$; and

2. similarly for $q$; as well as

3. the feature structure $F_{pq}$ whose path set is the prefix-closure of $\{p, q\}$,whose Nerode relation identifies only equal strings, and having $Z = \{(p, q)\}$.

By the finiteness of $A$, and the fact that $p$ and $q$ are finite strings, this set is finite.

For the inductive step, we use the reasoning just completed to check that there are finitely minimal $\models^-$-satisfiers of $f : \phi$ in the case of strong inappropriateness for $f$. Then for the case of $\neg\phi$, we notice that the minimal $\models^+$-satisfiers of $\phi \vee \psi$ are the minimal elements of the union of the set of minimal positive satisfiers of $\psi$ with the corresponding set for $\psi$, while the minimal positive satisfiers of $\phi\wedge\psi$ are obtained by pairwise unifying the minimal satisfiers of the components, and taking minimal elements. Then to get the negative satisfiers, we do the same calculations, reversing the roles of $\wedge$ and $\vee$. The remaining inductive cases are straightforward.

$\Box$

The corollary of all of this is that now we have a nonmonotonic feature logic which allows for negation, using the Smyth powerdomain semantics for the extended BCPO of negated and well-typed feature structures.

## *8.2 Inheritance*

In this last subsection we show how to use the finite partially ordered set of sorts as a way of prioritizing the construction of extensions. A similar scheme was suggested by Young [26], and the issue has been studied as well by Lascarides *et al.* [13]. Both these proposals are computationally oriented; unifications using Young's method of *non-monotonic sorts*, or Lascarides' notion of *persistent typed default unification*, can actually be carried out for lexical and other systems of defaults. Our proposal will in general be impossible to implement, though conceptually our priority scheme is rather simple.

Fix the sort hierarchy $(A, \sqsubseteq)$. This will now be called an inheritance hierarchy; the ordering is reversed from more standard treatments in that the most general sorts occur at the bottom, and the most specific sorts at the top. Next, associate with each sort $a$ a set $D(a)$ of default constraints. In a lexical hierarchy, for example, a constraint associated with the sort VERB might be that the past tense ending was "ed"; while for strong verbs such information would be overruled.

The problem to be solved is this: We start with a feature structure $F$ which has yet to be completed using constraints, both strict and default. We place $F$ into the hierarchy $A$ using the sort $a$ associated with its root. We then want to complete it in such a way that (i) it is resolved with respect to a fixed set $STR$ of strict constraints, and (ii) defaults associated with $a$ are applied first, and such that defaults associated with more specific sorts take priority over defaults associated with more general sorts. Of course, we only have to consider sorts which are more general than the initial sort $a$.

A simple scheme to solve this problem is the following. Start with the sort $a$. Partition the set $\downarrow a = \{b \mid b \sqsubseteq a\}$ as follows. Let $A_1 = \{a\}$; having defined $A_n$, let $A_{n+1}$ be the set $A_n$ unioned with the set of maximal (most specific) elements of $(\downarrow a) \setminus A_n$.

Next, define an increasing sequence of constraint sets $S(n)$ as follows. Let $S(0) = STR$, and for $n > 0$ put

$$S(n) = S(n-1) \cup \bigcup\{D(b) \mid b \in A_n\}.$$

Form the antichain default set $\Delta(n)$ associated with $S(n)$.

Now consider the following extension construction recipe.

- Let $E$ be the unique extension of the original structure $F$ using $\Delta(0)$. This is the set of defaults associated with $STR$; we assume here that the set of strict constraints does not lead to multiple default extensions. The singleton collection $E_0 = \{E\}$ is the collection of level 0 extensions.

- If the collection of extensions $E_n$ has been constructed, then for each $E \in E_n$ form all of its extensions using defaults from $\Delta(n)$. Call the resulting collection $E_{n+1}$.

The process stops when the most general sort $\perp$ has been considered. The *layered extensions* of $F$ are the elements of the final extension set, and can be considered the possible default completions of $F$.

Unfortunately, an extension constructed in this way will *not* automatically be resolved according to the strict constraints. However, $STR$ is included at each layer as a *default* set of constraints. Using the fact that extensions are closed under the *consistent* application of all default rules, this means that whenever a strict constraint is applicable to an extension, and the conclusion of the strict constraint is consistent with the extension, then the extension will "satisfy" the conclusion of the constraint.

More positively, no default set associated with a type will be applied before all the defaults associated with any of its more specific "subtypes" have been applied. Finally, at each stage each constructed extension will be (consistently) closed under all of the default rules "preceding" it in the layered scheme.

## 9. Conclusion

We summarize the main contributions of the paper, and we indicate further directions for research.

The principal technical result of the paper is Theorem 6.1. This shows that the antichain-update semantics we have chosen for defaults in the Smyth powerdomain is "correct" because we obtain the same kind of completion of a feature structure using disjunctive defaults as we do when we limit ourselves to purely conjunctive ones. The theorem does not by itself, however, guarantee unique default extensions. In the general case, when there is conflict between the "conclusions" of the default rules, we still can obtain them. Only in the case when there is a resolved element of the Smyth powerdomain dominating our starting element does the theorem guarantee unique extensions. Additionally, we might mention that our version of the Smyth powerdomain omits the top or inconsistent "empty" set. This differs from Pollard and Moshier [17], who make use of the Smyth domain as a complete distributive lattice. However, in the default setting, including the top element tends to trivialize the results, because we then never get multiple extensions; in cases like the Nixon Diamond it seems that such extensions are what is wanted.

More generally, we have indicated through the use of update defaults a universal way of adding disjunctive information to an element or elements of a domain which does not suffer from the "reasoning by cases" problem mentioned by Poole [19]. We plan to apply this idea in other Scott domains besides the domain of feature structures. A primary candidate is the domain-theoretic model theory for first-order logic studied in [22].

As problems needing further research, we should mention that our semantics for defaults yields a notion of nonmonotonic entailment only between compact elements of the Smyth powerdomain; these correspond in a natural way to the compact open sets in the Scott topology of the domain. We would like more generally to have a notion of entailment between arbitrary Scott open sets, since these correspond to "observable properties" of domain elements.

We also need to study the question of "irrelevant information" mentioned in Section 6.3. To show that our antichain semantics does not introduce such information, the primary problem is to say what we mean by the terminology.

A final problem, perhaps the most pressing, is to find a way to make our theory computationally realizable. At present one can only implement our definitions in finite spaces; even the space of feature structures will present a challenge if we are to use our present definitions. A start on the problem would be to reconcile the present results with the ideas on default unification presented in [26] and [13].

**Acknowledgements.** The present paper owes much to Mark Young, from whom we learned about nonmonotonicity. It also owes a debt to Bob Carpenter, and to the work of Carl Pollard and Drew Moshier. Support of the U.S. and Dutch National Science Foundations, and of CWI/Amsterdam, is gratefully acknowledged.

## References

1. P. Blackburn and E. Spaan. A modal perspective on the computational complexity of attribute value grammar. *Journal of Logic, Language, and Information*, 2:129–169, 1993.

2. P. Buneman, A. Jung, and A. Ohori. Using powerdomains to generalize relational data bases. *Theoretical Computer Science*, 91:23–55, 1991.

3. B. Carpenter. *The Logic of Typed Feature Structures.* Cambridge University Press, 1992.

4. A. Dawar and K. Vijay-Shanker. A three-valued interpretation of negation in feature structure descriptions. In *Proceedings of 27th Annual meeting of the Association for Computational Linguistics*, 1991.

5. G. Gazdar, E. Klein, G. Pullum, and I. Sag. *Generalized Phrase Structure Grammar.* Harvard university Press, 1985.

6. M. Gelfond, V. Lifschitz, H. Przymuzińska, and M. Truszczyński. Disjunctive defaults. In *Proceedings of Second Annual Conference on Knowledge Representation*, pages 230–237. Morgan Kaufmann, 1991.

7. C. Gunter. The mixed powerdomain. *Theoretical Computer Science*, 103:311–334, 1992.

8. C. A. Gunter and D. S. Scott. Semantic domains. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, volume B: Formal Models and Semantics*, chapter 12, pages 633–674. Elsevier, 1990.

9. M. Johnson. *Attribute-Value Logic and the Theory of Grammar.* Center for Study of Language and Information, 1988.

10. R. Kasper and W. Rounds. A logical semantics for feature structures. In *Proc. of 24th Meeting of the Association for Computational Linguistics*, pages 257–266, 1986.

11. R. Kasper and W. Rounds. The logic of unification in grammar. *Linguistics and Philosophy*, 13:33–58, 1990.

12. S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models, and cumulative logics. *Artificial Intelligence*, 44:167–207, 1990.

13. A. Lascarides, T. Briscoe, N. Asher, and A. Copestake. Order-independent and persistent typed default unification. To appear in *Linguistics and Philosophy.*

14. L. Libkin. *Aspects of Partial Information in Databases.* PhD thesis, University of Pennsylvania, 1994.

15. M. A. Moshier. *Extensions to Unification Grammar for the Description of Programming Languages.* PhD thesis, University of Michigan, 1988.

16. F. Pereira and S. Shieber. The semantics of grammar formalisms seen as computer languages. In *Proceedings of 10th International Conference on Computational Linguistics: COLING 84*, 1984.

17. C. Pollard and M. A. Moshier. Unifying partial descriptions of sets. In P. Hansen, editor, *Vancouver Studies in Cognitive Science: vol. I.* University of British Columbia Press, 1990.

18. C. Pollard and I. Sag. *Head-driven Phrase Structure Grammar.* University of Chicago Press, 1994.

19. D. L. Poole. What the lottery paradox tells us about default reasoning. In *Proceedings of First Annual Conference on Knowledge Representation.* Morgan Kaufmann, 1989.

20. M. Reape. *Introduction to Semantics of Unification-based Grammar Formalisms.* Kluwer, 1994.

21. Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

22. W. Rounds and G. Q. Zhang. Logical considerations on default semantics. To appear,*Proc. 3rd Int'l Symp. on Mathematics and AI*, 1994.

23. W. Rounds and G. Q. Zhang. Domain theory meets default logic. *Logic and Computation*, 5:1–25, 1995.

24. S. Shieber. The design of a computer language for linguistic information. In *Proceedings of 12th COLING*, pages 211–215, 1986.

25. M. Young and W. Rounds. A logical semantics for nonmonotonic feature structures. In *Proc. ACL Symp. on Computational Linguistcs*, 1993.

26. Mark Young. *Features, Unification, and Nonmonotonicity.* PhD thesis, University of Michigan, 1994.

27. G. Q. Zhang and W. Rounds. Defaults in domain theory. Submitted for publication.