Towards a Formal Specification for Defaults in GPSG

Roger Evans

September 1987

ɔgnitive Science Research Paper

:rial No. CSRP 084

:hool of Cognitive Sciences,
he University of Sussex,
:ighton BN1 9QN

Towards a Formal Specification for Defaults in GPSG

Roger Evans
University of Sussex
September 1987

Abstract

This paper examines the role of 'feature specification defaults" (FSDs) in the GPSG formalism and attempts to provide a characterisation of them in terms of formal non-monotonic logic. FSDs provide the grammar writer with a mechanism for specifying conditions on syntactic categories which should hold 'by default': a category must satisfy such a default condition unless specifically overruled by some other component of the grammar specification. The approach taken here combines recent work on the formal characterisation of syntactic categories *[G€izdar87]* with advances in the field of non-monotonic logic, in particular autoepistemic logic *[Moore83, Moore84],* to capture this default behaviour formally. In the process of this formalisation, a niunber of inadequacies and peculiarities of the GPSG account, both of defaults in particular and of syntactic structure in general, are revealed and discussed.

1 Introduction

A 'feature specification default' (FSD) is a statement in a GPSG[1] grammar to the effect that some condition on the structure of a syntactic category must hold for every category admitted by the grammar, unless some other component of the grammar specification overrides it. Thus the FSDs in a grammar describe the conditions on category structure that obtain "by default" - compare this with the 'feature co-occurrence restrictions' (FCRs) which describe conditions which necessarily obtain in all admitted categories. However, the account of FSDs given in [*GPSG85 p.100ff*] is technically complicated and obscure, and formally unsatisfying. In this paper we present an alternative formalisation of FSDs which is an improvement on both counts.

The account of FSDs developed below is based on recent work [*Gazdar87*] on the formal characterisation of syntactic categories. In particular we adopt a view of category descriptions and constraints (as one finds in grammatical rules) as statements in a 'category logic'. This leads naturally to the idea that default specifications might be statements in a non-monotonic extension to the same logic, thus allowing rule categories, FCRs *and* FSDs to be expressed in a single representation language. The particular brand of non-monotonic logic that we shall employ to implement this extension is 'autoepistemic' logic [*Moore83*, *Moore84*]. In this logic, the emphasis is placed on characterising complete axiom sets given a fixed set of assumptions (including default assumptions), rather than on reasoning by default, typically in a dynamic environment - in Moore's terminology, on autoepistemic, rather than default, reasoning. This makes it particularly appropriate in situations where the default behaviour is part of a purely declarative formal specification, as is the case here.

A reformulation of this kind necessarily involves considerable scrutiny of the existing mechanisms and their relationship to the rest of the GPSG formalism. This means that these proposals touch on wider issues for the formalism as a whole. However, as full discussion of these issues would involve virtually a complete description of the GPSG formalism, we provide only sufficient detail for the particular task in hand. Furthermore, the exact requirements of GPSG as specified in [*GPSG85*] pose some rather idiosyncratic problems when described in a more general setting. Hence we develop our formalisation in stages, dealing first with the simplest cases and then adding complexity in an attempt to reach the exact GPSG characterisation.

The principal goal of this work then, is a formulation of FSDs which is simpler than, but as far as possible equivalent to, the [*GPSG85*] account, is formally well-defined and declarative, and integrated with the category logic. But in addition to this we make some more general observations about the GPSG formalism as a whole, and about the role of defaults more generally in unification-based formalisms. However, it should be noted that this paper is only 'semi-formal'. Although a degree of formal rigour is maintained, the full technical details are omitted - these will be discussed in a companion paper.

2 Some Formal Preliminaries

The Category Logic L_c

We begin with an overview of the relevant parts of the category logic language L_c, of [*Gazdar87*] which we shall employ throughout. Only a brief description of the language itself is given, as the full details are worked out thoroughly in [*Gazdar87*].

The basic expressions of L_c of relevance to us are listed in (1). [2]

(1)	f	feature f is defined
	f:a	atom-valued feature f takes the value a
	f:ϕ	the constraint ϕ applies to the value of category-valued feature f
	$\neg\phi$	negation of a constraint
	ϕ and ψ	constraint conjunction
	ϕ or ψ	constraint disjunction
	$\phi \supset \psi$	constraint implication
	$\phi \equiv \psi$	constraint equivalence

In addition to these basic expression types, L_c includes two modal operators $\Box$ and $\Diamond$, whose principal role is in the delineation of a 'space of possible categories', (as discussed in [*Gazdar87*]). In our application of L_c as the language for specifying GPSG grammars, these operators never arise (at least, they are not needed for an account of the [*GPSG85*] formalism), and so we shall largely ignore their existence. However, they should not be confused with the default modal operators introduced below, which are quite distinct.

Sentences in L_c are interpreted as constraints on categories. [*Gazdar87*] defines a formal notion of a category (as a partial function from feature names to values) and provide a denotational semantics for L_c, essentially mapping sentences onto boolean valued functions over categories - 'tests' on categories that either succeed or fail. These functions accord with the informal descriptions provided in (1), and we need not be concerned with more formal detail here. If a category *C* is assigned the value 'true' by some sentence *S*, then we say *C satisfies S*. This notion of satisfaction is used to provide a straightforward definition of *modelhood* for sets of sentences in L_c.

(2) A category C is a model for a set of sentences T iff C satisfies each sentence in T.

Modelhood is the more convenient notion here because it recovers the notion of partial description much used in GPSG: a set of sentences can be viewed as a partial description of any of its models.

Mapping GPSG into L_c

The original purpose of L_c was to provide a language for describing 'category spaces' - sets of well-structured categories. [*Gazdar87*] mention in a footnote that the language is also suitable for representing the FCRs of a GPSG grammar. In this section we take this one step further, by proposing that L_c is appropriate, and indeed preferable, as a language for making all the statements about categories that arise in a GPSG grammar (here we omit FSDs, since that is the concern of the paper as a whole). We look in particular at category specifications in ID rules; extension to the categories in LP rules and meta-rules is straightforward [3].

Conventionally, GPSG grammars refer to categories in two ways. On the one hand we have categories that occur in ID rules, LP rules etc.. Formally speaking, these are instances of actual categories which may ultimately be part of parse trees. On the other hand we have FCRs which are partial descriptions of categories. These cannot themselves occur in parse trees, but constrain the categories that do.

Now FCRs can be viewed directly as sentences in L_c. But we can also embed actual category *instances* in L_c in a way which preserves the correct relationship between categories and constraints. We do this by mapping a category into a set of sentences of

L_c in a straightforward fashion; for each atom-valued feature/value pair in the category we include a feature/value constraint in the set, and for each category-valued feature/value pair we include path/value constraints for all the paths rooted in that feature. For example, the category in (3) (a) maps into the set of sentences in (3) (b).

(3) a) { <n,+>, <v,->, <bar,2>, <slash, {<n,+>, <bar,2>}> }
b) n: +
v: -
bar: 2
slash: n: +
slash: bar: 2

Using this mapping, we can replace an explicit category with a corresponding set of sentences in L_c. And we replace the notion of extension of a category (used in [*GPSG85*] in the construction of 'projections' of ID rules) with the notion of modelhood introduced above. Under this scheme an ID rule is a structure in which nodes are labelled with sets of sentences of L_c, and a projection of an ID rule is a corresponding structure where the nodes are labelled with models of the corresponding sentence sets.

This approach has several important advantages. First of all, it is formally tidier, since it reduces the number of distinct types of object in the GPSG language, by throwing away the explicit categories. Categories are now only referred to by virtue of being models of L_c, and thus lie solely in the semantics of the formalism, rather than in both syntax and semantics as was the case in [*GPSG85*].

Secondly, it allows us to view an individual category as a partial description of a category to which more complex constraints (such as FCRs) can simply be added. Thus instead of having to view FCRs as filters on legal categories, their effect can be achieved simply by adding them to the category descriptions themselves - legal categories must be models for the FCRs as well as for their individual category descriptions.

More generally, the feature instantiation principles (HFC, FFP etc.) may also be characterised in terms of judicious additions to category descriptions, rather than as filtering operations on local trees. We shall assume here that these principles operate in this way, by adding to the category descriptions provided by the ID rules. Adopting this technique for the principles clarifies their scope and their dependence on other components of the formalism, and makes issues of declarativeness more visible.

Finally, because this approach provides a uniform framework for representing all the category-specific statements, changes to the actual representation used (in this case the language L_c) permeate the whole formalism in a natural and consistent way. The case in hand is a good example: we will be able to extend L_c with modal operators to incorporate default behaviour without disturbing any other aspect of the formalism.

3 FSDs in GPSG

It was noted in the introduction that the role of FSDs in GPSG is somewhat complicated. In this section, then, we spend some time describing exactly what they are expected to do[4], and to a limited extent why they are as they are[5]. FSDs allow the grammar writer to make general statements about the structure of syntactic categories in default cases. For example, in a grammar of English one might want to say (4).

(4) "by default, *case* takes the value *ace**

This statement imposes a constraint on every category that is part of candidate syntax tree admitted by the grammar as a whole. Roughly speaking, the intended interpretation of the constraint is that if the rest of the grammar says nothing about the *case* feature on a particular category (so that it is 'free'), then it must take the value *ace.* If some other part of the grammar does constrain *case,* then the default can be ignored.

More generally, FSDs are arbitrary constraints on categories which must hold if the features they affect are free. These constraints are built out of basic feature-value constraints and the standard connectives of propositional logic *{and, or, not* etc.). In fact. we can view them as sentences of L_c [6]. (5) provides some sample FSDs using this constraint language.

(5) -*conj
case: ace
bar: 0 D -«vform: pas

The first of these states that by default, the *conj* feature is undefined (takes no value). The second is the formal statement of example (4), above. The last is a more complicated example: by default, if *bar* is *0* then *vform* is not *pas.* There are two ways this default can be satisfied, either by *bar* not being *0* or *vform* not being *pas.* Essentially this default says that the combination *bar: 0* and *vform: pas* is only permitted when specifically demanded by the grammar.

L_c provides an appropriate language for writing down FSDs, and indeed a semantics for them as boolean conditions on categories, but we still need a precise description of their role in the formalism as a whole. In particular, we need to specify how to tell in general whether the satisfaction of a default affects only 'free* features: if not, the default can safely be ignored. According to the *[GPSG85]* account, the appropriate conditions are given in (6) [7].

(6) A category in a local tree[8] need not satisfy a default D iff one of the following conditions is satisfied:
a) No local tree derived from the same ID rule satisfies D on the corresponding category.
b) Any local tree from the same ID rule that satisfies D on the corresponding category differs from the present local tree in some other category too.

The idea behind these criteria is as follows. If condition (a) holds, that is if all the other instantiations of the present ID rule also fail to satisfy the default, then we assume that something in the grammar is causing this, and so the default need not apply. Condition (b) covers cases where the default could apply, but in doing so some other category in the local tree would have to change too. This might happen, for example, if the default attempted to constrain a head feature, or an agreement feature. In such cases the default need not apply either. Roughly speaking, a feature is not 'free' if its value is constrained by the grammar, or forced by the grammar to covary with a feature in some other category.

An example may make the picture a little clearer. Consider the fragment of a GPSG grammar given in (7).

(7) **Features:** case {nom, ace} (a HEAD feature)
pro {+, -}

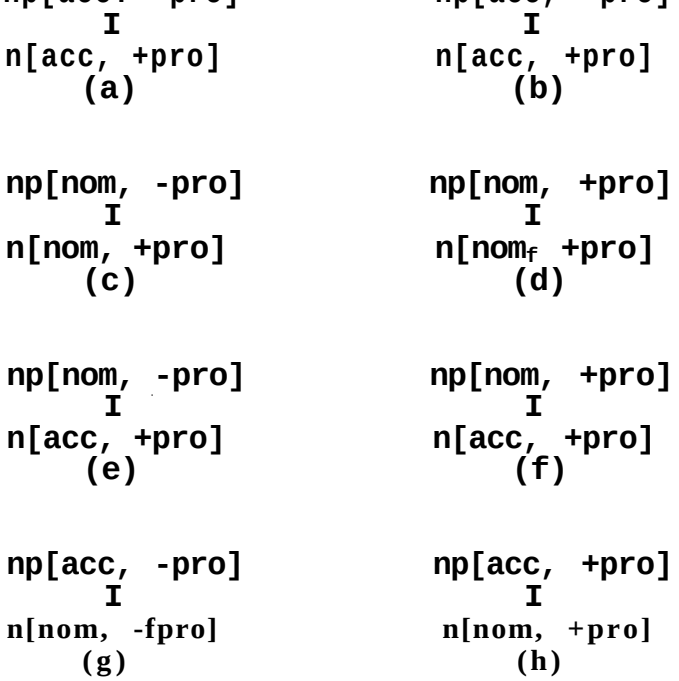
FSDs: case: ace
pro: -

ID rule: np -+ n[+pro]

Here we have two features: *case,* signifying nominative *(nom)* or accusative *(ace)* case marking, and *pro* signifying whether the category is pronominal (+) or not (-). *case* is a head feature, so its value is shared between mother and head daughter, while *pro* is not (mother and head daughter values are independent). Two FSDs are specified: *case* defaults to *ace, pro* defaults to -. Finally the single ID rule expands a noun-phrase as a single (head) daughter pronominal noun.

Instantiating the two features in all possible combinations on the ID rule gives the eight candidate local trees in (S).[9]

(8)



Four of these eight, (e,f,g and h), do not satisfy the head feature convention, which demands that the *case* features on mother and daughter coincide, and so they can be ruled out of consideration immediately. Which of the remaining four are acceptable according to the FSDs? Looking first at the *pro* feature on (a), the mother value satisfies its default but the daughter value does not. However none of the daughter values do (in any of the trees a.b,c and d) so (6) clause (a) applies and the default can be ignored. So (a) is acceptable as far as *pro* is concerned. In (b) on the other hand, the mother *pro* feature violates the default and this time there are local trees in which it is satisfied (a and c). Furthermore (a) is otherwise identical to (b). so neither clause of (6) applies. Thus (b) is not acceptable. Similarly (c) is acceptable and (d) is not.

Turning to the *case* feature, we see first that (a) is acceptable because both categories satisfy the default anyway. Consider now the mother category of (c). This violates the default for *case* and there are local trees (a and b) in which the default is satisfied, so

(6) clause (a) cannot apply. However in both those trees the daughter category is also different from (c), so (6) clause (b) does apply. Exactly the same reasoning can be applied to the *case* feature on the daughter of (c) as well. Hence we see that of the eight trees exactly two (a) and (c) are admissible in a GPSG syntax tree.

4 FSDs in Isolated Categories

We begin our reformulation of GPSG defaults by looking at a substantially simplified case, obtained by imposing an additional constraint on the grammar formalism: we assume that categories in local trees are entirely independent of each other. In other words no grammatical constraints on a category depend on other categories in the local tree. This simplification allows us to treat categories essentially in isolation - the other categories of the local tree are irrelevant. We exploit this by ignoring local trees entirely, and thinking of grammars solely in terms of the categories they characterise, regardless of the configurations of those categories in local trees. Among other things, this allows us to ignore the second clause of (6) completely, because the situation it describes can only arise when there is inter-category dependence. Thus for the time being, our characterisation of defaults will be (9).

(9) A category derived from an ID rule need not satisfy a default D iff no other corresponding category derived from the same ID rule satisfies D.

(9) is expressed in terms of the [*GPSG85*] notion of categories in ID rules. Let us look at what it says when translated into the L_C version of GPSG introduced above. We have some category description *C* encoding information from an ID rule, *R*, the FCRs and all the principles. We also have an FSD *D* (a sentence of L_C). Since *C* contains all the constraints of the grammar (apart from the FSDs), and the categories in local trees are independent, every possible model of *C* occurs as the 'corresponding category' in some local tree under consideration in (9) (there being no other factor to constrain candidate models of *C*). Thus (9) states that *D* can be violated in *C* iff no model of *C* satisfies *D*, or, by a basic theorem of model theory, iff *D* is inconsistent with *C*. Otherwise *D* must be satisfied.

This suggests the following alternative characterisation of the role of an FSD.

(10) Given a category description C, and an FSD D we obtain a new description C' which takes account of D as follows:
a) $C' = C \cup \{D\}$ if D is consistent with C
b) $C' = C$ otherwise

This proposal has a number of advantages and disadvantages. On the positive side, it is simple, precise and avoids examining all the possible models for *C*. On the negative side it is not declarative: it involves 'doing something' to the category description. It is to overcome this latter objection that we turn to autoepistemic logic.

5 Using Autoepistemic Logic

Autoepistemic logic was introduced by Moore [*Moore83, Moore84*] as a formalism for characterising 'autoepistemic' reasoning, or "reasoning about one's own knowledge or belief" ([*Moore83 p.273*]). He extends the predicate calculus with a modal operator L which is read as 'is believed'. This introduces a class of new expressions such as (11).

(11) a) L(P) P is believed *or* I believe P
b) L(L(P) D P) I believe that if I believe P then P is true

The idea is that given a set of sentences *T* in this extended logic we can ask "What are the logical consequences (for me) of the fact that I believe all the sentences in T?". Alternatively, what is my belief set, given only that I believe *T,* and assuming I believe everything in my belief set and do not believe anything outside my belief set and I am capable of perfect logical reasoning. Such a belief set is called a *stable expansion* of *T,* and is defined more formally in (12).

(12) Given a set of sentences T, a stable expansion T of T is a set of sentences such that:
a) T is tautologically closed
b) If P € T then L(P) € *T*
c) If P 5 in T then -*L(P) € *T*
d) every sentence in T* is a tautological consequence of
T U {L(P): P € T'} n {-JL(P): P 5 T'}

Stable expansions are not unique (a given set of sentences may have more than one stable expansion) but they have two important logical properties. Any stable expansion of a set of sentences *T* is sound (with respect to *t)* and complete. Thus treating *T* as a set of 'axioms', a stable expansion of *T* can sensibly be viewed as a complete set of theorems derived from those axioms.

To see how this is relevant to GPSG defaults, we need to incorporate these ideas into L_c. Observe first of all that we can extend L_c with the modal operator L in much the same way as Moore extends predicate calculus, giving us a new language L^, and import the notion of a stable expansion for a set of sentences of L^ [10]. In doing so, however, it seems sensible to move away from the anthropological terminology of 'belief to the more neutral conventional modal notion of 'necessity'[11]. And in fact, it is more appropriate for us to use the dual modal operator M as defined in (13).

(13) M(P) s -L(-*) P is possible

Thus $L_{\_\_}$ contains sentences such as (14).

(14) M(case: ace) It is possible for *case* to be *ace*
M(-*conj) D -»conj If it is possible for *conj* to be undefined, then it is undefined

It is sentences of this second form, M(P) D P, which are of particular interest because they possess the following property. Let P be a nonmodal sentence of L_{CT} (that is, P does not containing any instances of the modal operators, L or M). Suppose M(P) D P is in some set of sentences, *T,* and let *T* be any stable expansion of *T.* Then P is in *T* iff -»P is not derivable from *T.* Or, P is in T iff P is consistent with *T.* To see this informally, notice that M(P) D P has the reading "if P is possible then P is true". P will be possible unless some other basic axiom contradicts P, in other words, unless P is inconsistent with the given axioms. Thus if M(P) D P is itself an axiom, its antecedent will be true when P is not inconsistent with the axioms, and so its consequent, P, will be derivable in just those cases.

This property of such modal expressions allows us to reformulate (10) in a more satisfactory fashion. Recall that we have been viewing the category specifications in grammars as sets of sentences of $L_{\_}$, whose models are the actual categories represented. Recall also that we noted that FCRs can be accommodated simply by adding them (as

sentences in L_C) to the category descriptions. Suppose now we take category specifications to be sets of sentences of L_{CE} (which contains L_C), and define a model of a set of sentences of L_{CE} to be an L_C-model of the non-modal sentences in a stable expansion of the set[12]. In other words getting from category description to category is now a two-stage process: from category description to stable expansion, from stable expansion to L_C-model (a category). This change will have no effect on sentences which contain no modal operators, so our existing pieces of formalism will remain undisturbed. But the characterisation of FSDs is now (15).

(15) For each FSD D, add $M(D) \supset D$ to every category description.

This proposal puts FSDs on exactly the same footing as FCRs. Indeed one can now view FSDs as simply a special case of FCRs which happen to include the modal possibility operator. Furthermore their application is entirely declarative - they are simply another component that goes to make up a category description [13].

6 FSDs in Covarying Categories

So far, we have been considering categories only in isolation. For this approach to be practically useful, in GPSG or other related formalisms, we need to extend it to allow inter-category dependencies. In this section, we take the next step by considering the general case obtained simply by adding mechanisms for inter-category dependencies. Later we shall look at the more constraining requirements of the actual GPSG case.

First of all, here are two revealing observations about the [*GPSG85*] formalism. The first is that it is not possible to specify inter-category dependencies in the grammar itself, beyond the simple local tree relationships in ID and LP rules[14]. All such dependencies arise through the application of feature instantiation principles such as HFC. The second is that the much of the complexity of the feature instantiation principles (and especially the application of FSDs) arises from the use of properties of entire local trees and the attendant need to quantify over sets of 'candidate projections'. These two phenomena are related in that the first is essentially the cause of the second. Because inter-category dependencies are not supposed to expressible in the grammar itself, the formalisation of GPSG in [*GPSG85*] does not include mechanisms for expressing them. And because it lacks such mechanisms, the specification of the feature instantiation principles (which could make good use of them) is unmanageably complex.

Two new formal mechanisms are needed to deal with inter-category dependencies. The first is a formal language for describing whole local trees, rather than just categories (as provided by L_{CE}). This is necessary since it is in the context of a local tree that such dependencise arise. The second is a way of stating dependencies in such a language[15].

For the first of these mechanisms, we introduce the language L_{LT}, a local-tree description language based on L_{CE}. Only an outline of the structure of L_{LT} is given here; in particular we omit a full formal specification of either the syntax or the semantics. In addition, as a local tree language for GPSG, L_{LT} should be able to represent the constraints imposed by ID rules, LP rules etc. independently. However in the following discussion we omit discussion of how this might be done, and assume a simple context-free specification for the basic local tree skeleton.

The objects in L_{LT} consist of the objects in L_{CE}, augmented by a disjoint set C of

'category variables'. Sentences of L_{LT} come in the following forms:

(16) a) $A \rightarrow B_1, \ldots, B_n$ where A, B_i are in C.
b) $A \mid \phi$ where A is in C, ϕ is in L_{CE}.
c) ϕ where ϕ is in L_{CE}.

A set of sentences form a description of a local tree. The definition of model of L_{LT} is slightly more involved than was the case with L_C. The idea is that sentences such as (a) define the basic local tree structure, identifying category variables with actual categories in the local tree. Sentences such as (b) specify constraints on individual categories in the tree, and those like (c) specify constraints on all categories in the tree. To capture these intentions, we define a model as follows.

(17) Let C be the set of category variables in L_{LT}. For each C in C, define a projection mapping P_C from sets of sentences in L_{LT} to sets of sentences in L_{CE} as follows:

$$P_C(T) = \{\phi, \phi \in L_{CE} \cap T\} \cup \{\phi, C \mid \phi \in T\}$$

Now let T be a particular set of sentences in L_{LT}. A local tree L is a model for T relative to a mapping I from C to the labels of L iff
a) if $A \rightarrow B_1, \ldots, B_n \in T$ then I(A) labels the mother of L, and $I(B_i)$ labels the i'th daughter of L.
b) for all c, I(c) is an L_{CE}-model of $P_C(T)$

The projection functions P_C pick out the sentences in the description that concern a particular category variable *C*. A model is a local tree equipped with a mapping from category variables to actual categories in the tree that conforms to the 'context-free rule' sentences (there will usually be only one such in a description, but if there are many, they all have to be satisfied), and such that each target category is a model (in the L_{CE} sense - an L_C-model for a stable expansion of the sentence set) for the sentences that concern it.

Notice that we have made specific allowance for statements that apply to all the categories in a local tree. Generally speaking, FCRs and FSDs will appear as statements of this form, rather than being repeatedly enumerated for each category, as our earlier exposition would suggest. Thus the formalism recovers something of the global nature of FCRs and FSDs[16].

Before providing an example of a local tree description, we shall add the other of our new mechanisms to the formalism, namely a way of specifying inter-category constraints. We do this by extending the underlying category language, L_C with feature variables, a new set of tokens, distinct from features, feature values and category variables, which may occur in place of actual values in sentences of L_C [17]. So let V be this set of feature variables. We extend L_C by allowing expressions of the form:

(18) f: V V in V, f an atomic- or category-valued [18] feature

We call this extended language L_C^v, and extend our notion of modelhood to take account of these variables as follows. Given a set of sentences, *T*, of L_C^v, a category *C* is a model of *T* relative to a mapping *S* from V to feature values in *C* iff all the sentences of *T* are satisfied where

(19) f: V is satisfied iff C(f) - S(V)

(Recall here that categories are partial functions from features to values). Thus 5 provides values for the feature variables in *T,* which C has to satisfy[19]. This straightforward extension can be imported unproblematically into L^, giving L^. and from there into L_{LT}, giving $L^*_{L\,r}$ A minor complication in this latter stage arises from the fact that we want a single substitution function for all the categories in an $L£_T$ description. Thus our notion of an $L^*_{L\,T}$ model is (20) (with C, P_c, V as above).

(20) Let T be a set of sentences in $L^*_{L\,r}$ A local tree L is a model for T relative to a mapping S from V to values in the labels of L and a mapping I from C to the labels of L iff
a) if A -> B_r •… B_n € T then I(A) labels the mother of L, and I(B_f) labels the i'th daughter of L.
b) for all c, I(c) is an L^-model of P_C(T) relative to S

7 An Example of a Local Tree Description

Before turning to more specific GPSG issues, here is an example of how this local tree language works. Consider once more the GPSG grammar fragment of (7), reproduced as (21).

(21) np -• n[+pro]

fsd: case: ace
fsd: pro: -

Rewritten as a local tree description in $L^*_{L\,r}$ this is (22).

(22) A -> B
A 1 cat: np
B I cat: n
B 1 pro: +
M(case: ace) D case: ace
M(pro: -) D pro: -

Notice here the M(P) D P expansion for the two FSDs, and also that the FSDs are global constraints. Application of the HFC leads to an inter-category constraint, which is achieved by the use of a variable (V)[20], leading to (23).

(23) A -> B
A ! cat: np
B I cat: n
B I pro: +
M(case: ace) ^ case: ace
M(pro: -) D pro: -
A I case: V
B ! case: V

For a local tree to be a model for (23), there must be a mapping from the category variables *A* and *B* to categories labelling the nodes in the local tree such that *A* maps to the mother and *B* maps to the unique daughter and a mapping from V to some feature value such that the mother is an LJ^-model for (24) (a) (which is the projection of (23) onto A), and the daughter is an L^-model for (24) (b).

(24) a) cat: np
M(case: acc) ⊃ case: acc
M(pro: -) ⊃ pro: -
case: V

b) cat: n
pro: +
M(case: acc) ⊃ case: acc
M(pro: -) ⊃ pro: -
case: V

Such L^v_{CE}-models will be L^v_C-models of stable expansions of the above descriptions. So we are interested in what additional sentences of L^v_C (that is, non-modal sentences) are in such stable expansions. Looking at description (a) first, both *case: acc* and *pro:* - are consistent with (a), so M(*case: acc*) and M(*pro:* -) will be true in the stable expansion. Thus *case: acc* and *pro:* - will be in the stable expansion. Turning to (b), *case: acc* will similarly be in the expansion, but *pro:* - is inconsistent with (b) (because *pro:* + is in it), so M(*pro:* -) will be false and the default sentence will have no effect. Thus (25) is a model for (22) relative to a substitution function that maps *V* to *case*.

(25)
```
np[acc,-pro]
      |
n[acc, +pro]
```

8 The GPSG case

The treatment proposed so far may be an appropriate model of default behaviour for some theories of syntax, but it fails to capture the GPSG case in detail. For example, if we assume there are no other features in our grammar, then (25) is the only L^v_{LT} model of (22). But looking back at the discussion in section 3, we see that GPSG requires that a nominative version of this local tree should also be acceptable; clause (b) of (6) allows it by using the covariation between categories to block the application of the default.

In our new version of the formalism, covariation can only arise through the explicit use of feature variables (we do not allow any 'hidden' covariation introduced contingently by feature instantiation principles). This means that we can recover the effect of clause (b) by taking note of variable assignments in our models in the following way. Suppose we have a local tree containing a category in which some feature violates its default, and there is some other local tree derived from the same local tree description in which the default is satisfied. The violation is only acceptable if a covariation causes some other category in this second tree to be different also. This can only happen if the variable assignments used in the two trees differ. Thus to allow for the covariation case we need to say that a violation is only unacceptable if there as some other local tree *with the same variable assignment function* that obeys the default.

In order to characterise this formally, we need to look briefly at the semantics of autoepistemic logic. [*Moore84*] defines a 'possible-world' semantics for autoepistemic logic, in which stable expansions can be modelled by a set of possible worlds (corresponding to the set of possible current worlds consistent with the agent's beliefs) and a single actual world (corresponding to the way the world actually is). Such models can be used to provide a semantic definition for the modal operators. Suppose T is a set of sentences of L_{CE}, W is a set of worlds and A a particular world. The pair <W,A> defines an autoepistemic interpretation of the sentences of T, as given in (26).

(26) a) If P is nonmodal, then P is true iff P is true in A.
b) If P is nonmodal, then L(P) is true iff P is true in every world in W.
c) The truth of more complex modal expressions is derived in the conventional truth-functional fashion.

Clause (b) defines the modal operator L semantically: L(P) 'means' "P is true in every believable world". Such an interpretation is an autoepistemic model for T iff all the sentences in T are true[21].

In this definition, the worlds (those in W and the actual world A) are propositional interpretations for T: sentences in T must be true or false in them. Recall now that in our extended logics, the definition of such a propositional model depends on a substitution function to provide the values for variables. Thus when we import the above notion of a possible worlds model to L^{v}_{CE}, A and each world in W has associated with it a substitution function. We shall call a world an *S-world* if *S* is the substitution function associated with it.

Suppose now we have a pair <W,A> as before, where A is an S-world for some substitution function S. We can redefine the modal operator L to be sensitive to variable substitution changes, by replacing clause (b) above with (27).

(27) If P is nonmodal, then L(P) is true iff P is true in every S-world in W.

Expressed in terms of the dual modal operator M, this is (28).

(28) If P is nonmodal, then M(P) is true iff there is some S-world in which P is true.

Consider what ¬M(P) means in this system. *S* is the substitution function of the actual world, the 'current' substitution function. ¬M(P) will be true iff P is not true in any *S*-world, in other words, in order to make P true (if it is possible at all), you have to change the substitution function. When we compare this latter statement with (6) clause (b), we see that here we have 'to make P true you have to change the substitution function' where there we had, in effect 'to make P true you have to change some other category'. But we have already noted that the only way it could be essential to change some other category is if it covaries in some way with the current one (otherwise it would be possible to change the current one without affecting the other one). Since the only way covariation can be expressed is via variables and the substitution function, any change to another category must be due to a change in some variable value, that is, a change in the substitution function. So this new notion captures the intent of the original GPSG one[22].

If we revisit the example of the preceding section, we see how this new definition of M works. Suppose we have a candidate model local tree in which *case: nom* is specified. Since the *case* features in the description have variable values, *nom* must be specified by the current substitution function. Now if we examine whether M(*case: acc*) is true in this model we see that it is not - there are no models with the same substitution function in which *case* is *acc* (since the only thing the function does is assign *nom* to *case*). Thus M(*case: acc*) fails and so the nominative case specification is acceptable.

This proposal, then, brings us somewhat closer to the requirements of the GPSG notion of defaults, and indeed captures the right behaviour exactly in the example case. However, it is somewhat unsatisfactory in its reliance on a semantic definition of the modal operator with no corresponding syntactic counterpart. Given a candidate model, we can decide whether it is indeed a model, but there seems no obvious syntactic way

of characterising a 'stable expansion* for a theory with our new interpretation of the modal operator. The root of the problem is the interaction between the modal operators and the variable substitutions: the modal operator is sensitive to variable substitutions, but the latter are part of the semantics of the language L^*_{τ}. Thus the modal operator can only be defined semantically. To alleviate this problem, then, we need either a syntactic notion of variable substitution, or an interpretation of the modal operator that does not depend on variables.

9 Conclusions

This paper has developed an alternative approach to the semantics of FSDs in GPSG. culminating in a proposal which captures much of the content of the *[GPSG85]* proposal. Along the way, we have argued the case for using descriptions of categories rather than categories themselves in grammar specifications, and for properly defining a language for describing local trees and inter-category dependencies. Adopting these proposals would make for a far cleaner and clearer formalism, quite independently of their particular use here.

There remain many outstanding issues and questions raised by these proposals, of which the more immediate are offered below.

- How close is the above approach to the GPSG account in fact?
- Is there a better characterisation of the desired modal behaviour than was presented in section 8, and in particular a syntactic one?
- Should the GPSG notion of defaults be changed to fit in more neatly with a modal scheme similar to that proposed above?
- Can these modal operators be used to represent other aspects of the GPSG formalism (eg the default-like behaviour of HFC)?
- What other similar mechanisms might be proposed to accommodate other aspects of GPSG - are there neat declarative formulations of the feature instantiation principles?
- Can $L£_T$ be extended to cover the whole of GPSG - to actually replace the GPSG formalism itself?
- If so, to what extend can the restrictiveness objectives of GPSG be incorporated into the formalism?

Footnotes

1. **Generalised Phrase Structure Grammar - see *[GPSG85]* etc..**

2. **Notice here that one can demand that a feature takes a value, without specifying any particular value; GPSG categories are partial mappings from features to values and so in general a given feature need not take any value at all.**

3. **See also *[KilburyS6]* for proposals on similar lines.**

4. **We make one simplification to the *[GPSG85]* treatment here, by ignoring the special case of lexical head daughters discussed on p.103. See also *[Warner87],* which proposes an alternative treatment for the phenomena covered by this special case.**

5. **For further discussion on what is required of default behaviour see *[GPSG85]* ch.2 and 5, and *[Gazdar&Sl***

6. [*GPSG85*] uses a different notation but translation is trivial.

7. These conditions do not correspond exactly to the expectations of the informal description given above: for example if the grammar specifies that *case* should be defined (via an FCR, say), but does not specify a value, then the default may still have to apply, even though one might have though *case* was no longer 'free'. However it is the more specific [*GPSG85*] criteria that we shall assume below.

8. Strictly speaking, the local trees under consideration here are only those that satisfy HFC, FFP, CAP etc. See [*GPSG85*] ch.5 for full details. We shall assume in all the following definitions relating to defaults that a similar constraint applies, unless we specifically state otherwise.

9. Here we adopt a fairly traditional notation for categories, rather than the somewhat unwieldy fully formal representation.

10. Actually this statement may depend on whether L_C has nice logical properties. Jeremy Carroll has pointed out that without infinite categories L_C is not compact. However it appears that if infinite categories are allowed in L_C then it is possible to embed it into first order predicate calculus.

11. We must exercise caution here, because this necessity is relative to an agent's belief system, rather than a set of 'possible worlds' in the conventional sense. Thus the statements L(P) is not read as "P is necessarily true (in all possible worlds)" but rather as "P is necessarily true in all my models of the way the world is". The fact that L(P) holds says nothing about the value of P in worlds other than the actual one (autoepistemic logic does not directly address the ordinary notion of possible worlds at all), and in fact L(P) $\supset$ P need not be valid (an imperfect agent may believe something that is not true).

12. [*Moore83*] theorem 1 ensures that this definition corresponds to the standard notion of an autoepistemic model.

13. In fact, we could simplify this proposal still further, by defining an operator D by D(P) $\equiv$ M(D) $\supset$ D. Then for each default D, one would add just D(D) to a category description. If in addition we ban the explicit use of L and M from L_{CF}, we can reduce the expressive power of the language without any linguistic penalty. However, this point is not pursued here.

14. Actually this statement is untrue: the use of rule schemas in [*GPSG85*] does allow one to specify such dependencies. However, no formal account for such schemas is provided in [*GPSG85*] and we assume these to be an informal notational device, and hence not part of the formalism itself.

15. It is relevant to note here that PATR-2 [*Shieber83*] has both these mechanisms, although not formulated identically. This is a major factor in the elegance of the translation of GPSG into PATR-2 described in [*Shieber86*]. However, the semantic account of local trees in [*Pereira84*] treats them as something of an afterthought, rather than a substantial component of the formalism in their own right.

16. In fact, of course, these rules should be global to the local tree descriptions as well, and to capture this one would have to reformulate the whole of a GPSG grammar as a single description of partial tree descriptions, which is beyond the scope of this paper.

17. PATR-2 achieves the same effect with equational constraints, [*Kasper86*] with equivalence classes. Our choice of feature variables here is mainly for convenience in later definitions.

18. In fact, to fully accommodate GPSG, we would probably want category-valued feature variables and category variables (in C) to be the same, but for simplicity we ignore this point here.

19. A comment about [*Gazdar87*] section 7 is relevant here. They discuss the problems posed by categories having 'shared values' - a single value occurring in two places in a category. Our view of variables here can be taken as meaning either a single actual object occurs in many positions in a category, or that distinct but identical copies of an object do - it depends on the interpretation of '=' in (20). The problems they discuss arise from the attempt to manipulate categories directly (for example, to unify them), as [*GPSG85*] does. In the approach advocated here, where only category descriptions are manipulated, this issue seems something of a red herring, the (clearly important) distinction between contingently equal and shared values can be adequately captured at this level and is easily formalised, as we do here.

20. As mentioned above, we assume HFC operates by extending the local tree description in this way, although we do not currently have any mechanism for achieving this.

21. The notion of an L_{CE}-model used above fits in here as follows. A is an L_{CE}-model for a theory T iff there exists a set W such that <W,A> is an autoepistemic possible-worlds model for T.

22. In fact it is more general in two ways: it will detect covariation within a single category and it may also pick up 'redundant' uses of variables - variables that occur in only one place in a local tree description. Neither of these cases occurs in the [*GPSG85*] formalism so we may either arbitrarily ban them (in well-formed GPSG local tree descriptions) or put up with their consequences.

References

GPSG85 Gerald Gazdar, Ewan Klein, Geoffrey Pullum and Ivan Sag. *Generalized Phrase Structure Grammar*, Blackwell, Oxford, 1985.

Gazdar85 Gerald Gazdar, "Linguistic applications of default inheritance mechanisms," in *Proceedings of the Alvey/ICL Workshop on Linguistic Theory and Computer Applications*, ed. Peter Whitelock, Harold Somers, Paul Bennett, Rod L. Johnson and Mary McGee Wood, pp. 27-48, CCL/UMIST, Manchester, 1985.

Gazdar87 Gerald Gazdar, Geoffrey K. Pullum, Robert Carpenter, Ewan Klein, Thomas E. Hukari and Robert D. Levine, "Category structures," Report No. CSLI-87-102, Center for the Study of Language and Information, 1987. To appear in Computational Linguistics, Volume 14, 1988

Kasper86 Robert Kasper and William Rounds, "A logical semantics for feature structures," *ACL Proceedings, 24th Annual Meeting*, pp. 257-266, 1986.

Kilbury86 James Kilbury, "Category cooccurrence restriction and the elimination of metarules," *COLING-86*, pp. 50-55, 1986.

Moore83 Robert C. Moore, "Semantical considerations on nonmonotonic logic," *IJCAI-83*, pp. 272-279, 1983.

Moore84 Robert C. Moore, "Possible-world semantics for autoepistemic logic," *Workshop on Non-Monotonic Reasoning, New Paltz, New York*, pp. 344-354, 1984.

Pereira84 Fernando C.N. Pereira and Stuart M. Shieber, "The semantics of grammar formalisms seen as computer languages," *COLING-84*, pp. 123-129, 1984.

Shieber83 Stuart Shieber, Hans Uszkoreit, Fernando C.N. Pereira, Jane J. Robinson and M. Tyson, "The Formalism and Implementation of PATR-II," in *Research on Interactive Acquisition and Use of Knowledge*, SRI International, Menlo Park, Ca., 1983.

Shieber86 Stuart M. Shieber, "A simple reconstruction of GPSG," *COLING-86*, pp. 211-215, 1986.

Warner87 Anthony R. Warner, "Simplifying lexical defaults in generalized phrase structure grammar," *Linguistics*, vol. 25, no. 2, pp. 333-340, 1987.