A Feature-Based Syntax/Semantics Interface

John Nerbonne *
Deutsches Forschungszentrum für Künstliche Intelligenz
Stuhlsatzenhausweg 3
D-6600 Saarbrücken 11, Germany
nerbonne@dfki.uni-sb.de

Abstract

Syntax/Semantics interfaces using unification-based or feature-based formalisms are increasingly common in the existing computational linguistics literature. The primary reason for attempting to specify a syntax/semantics interface in feature structures is that it harmonizes so well with the way in which syntax is now normally described; this close harmony means that syntactic and semantic processing (and indeed other processing, see below) can be as tightly coupled as one wishes—indeed, there need not be any fundamental distinction between them at all. In this paper, we first point out several advantages of the unification-based view of the syntax/semantics interface over standard views. These include (i) a more flexible relation to nonsyntactic constraints on semantics; (ii) a characterization of semantic ambiguity, which in turn provides a framework in which to describe disambiguation, and (iii) the opportunity to underspecify meanings in a way difficult to reconcile with other views. The last point is illustrated with an application to the notorious scope ambiguity problem.

Keywords: Computational Linguistics, Semantics, Constraint-Based, Compositionality

Contents

*Thanks to Bob Carpenter, Kris Halvorsen, Bob Kasper, Walter Kasper, Uli Krieger, Joachim Laubsch, Nissim Francez, Carl Pollard, Peter Ruhrberg and two anonymous *Mathematics of Language* referees for discussion of the ideas presented here. This work was supported by research grant ITW 9002 0 from the German Bundesministerium für Forschung und Technologie to the DFKI DISCO project.

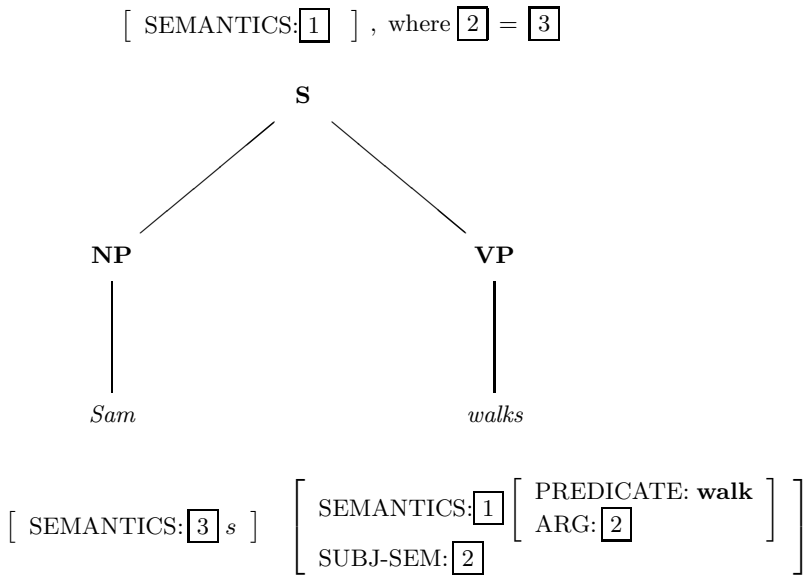


Figure 1: A sketch of the semantic derivation of *Sam walks*, **walk** (s) as this would proceed using unification. Unification applies to syntactic and semantic representations alike, eliminating the need to compute these in distinct processes, and unification is employed to bind variables to argument positions, eliminating the need for (a great deal of) β-reduction as used in schemes derived from Montague and the lambda calculus.

1 Introduction

"Standard" natural language algorithms and processing techniques were ill-equipped to deal with feature-based grammars; this difficulty gave rise to "unification-based" formalisms and processing models, which systematize the processing and even the theory of feature-based grammars. The application and development of unification-based approaches quickly demonstrated additional virtues, e.g., the ability to cater to the information shared in linguistic signs, and the ability to exploit partial information.

Syntax/Semantics interfaces using unification-based or feature-based formalisms may be found in Shieber 1986, Pollard and Sag 1987, Fenstad et al. 1987, and Moore 1989. The primary reason for specifying a syntax/semantics interface in feature structures is that it harmonizes so well with the way in which syntax is now normally described; this close harmony means that syntactic and semantic processing (and indeed other processing, see below) can be as tightly coupled as one wishes—indeed, there needn't be any fundamental distinction between them at all. In feature-based formalisms, the structure shared among syntactic and semantic values constitutes the interface in the only sense in which this exists.

1.1 A Simple Illustration

The fundamental idea is simply to use feature structures to represent semantics. If one wishes to compute the semantics of a sentence such as *Sam walks*, one first defines a feature SEMANTICS which must be lexically provided for in the case of *Sam* and *walks*, and which can be computed from these (together with other information) in the case of the sentence. Figure 1 provides an illustration of how this works.
Nerbonne 1992a

explores the advantages of the unification-based view of the syntax/semantics interface over standard views, arguing that it is superior first in allowing the statement of nonsyntactic constraints on semantics, e.g. those arising from phonology or contextual pragmatics; second in enabling the statement of constraints which generalize over syntax and semantics, such as the Subcategorization Principle or the binding inheritance principle of hpsg, Pollard and Sag 1987 (The computation shown in Figure 1 is a consequence of the Subcategorization Principle); third, in providing a level of constraints on logical form at which disambiguation constraints may be applied; and fourth, in allowing the underspecification of meanings in ways difficult to reconcile with non-constraint-based accounts. In this final category, the example was scope ambiguity. The present paper contributes to the mathematics of the relation between feature descriptions and semantic interpretations by providing a rigorous account of scopally underspecified semantics in terms of feature structure descriptions.

It is appropriate to review the third point—the ability to characterize ambiguity within feature description languages—in this introductory section as well, since it clarifies the nature of the undertaking, and since it has proven controversial. This is the subject of the section below.

The view defended here is that semantic features (directly) describe meanings, not model structures. For the sake of concreteness we shall treat meanings as the expressions in a logical language. Our view can then be made crisper: semantic features (directly) describe logical forms, not model structures.

It is important to note that this is not a merely philosophical point, whose answer nothing of practical interest turns on. On the contrary, a proper understanding of ambiguity enables the treatment of disambiguation, which is required in virtually every natural language understanding (NLU) task—and NLU is still the most important application of computational semantics. The importance of disambiguation extends beyond NLU, however, to virtually every application of computational semantics: disambiguation and the accompanying filtering of analytical hypotheses is the primary motivation for including semantics in speech and handwriting recognition efforts.

2 What does Feature-Based Semantics Describe?

But the primary reason for posing the question here is internal. We need clarity about the nature of semantic descriptions—the value of [SEMANTICS] attributes—in order to push forward to the more challenging tasks in semantic representation and reasoning. For example, we need answers to the following questions:

- How are quantifiers, and more generally, higher-order operators, to be represented?
- How can lambda expressions be represented? If they are inexpressible, how can one treat those areas where lambda expressions seem to be employed crucially—type-raising (*The CEO and every division manager left*), VP and common noun phrase ellipsis (*Tom read every book before Sam did* and *A good big man will beat a small one*) and predicative traces (*How tall is Sam?, How tall did Al say that Sam is?*)?
- There exist standard model theories for feature formalisms (of somewhat different sorts—cf. Kasper and Rounds 1986, Smolka 1988 and Carpenter et al. 1991) Are these sufficient for the use of feature structures to represent natural language semantics?

There is a simplest view of the nature of feature-based semantic representations, which is that they may be interpreted as directly denoting elements within a semantic model. I shall contrast this with the view that feature structures are better used to describe, not model structures directly, but rather logical forms (which in turn denote elements within the model structure). Figure 2 provides a graphic rendition of the second, more complicated view, which shall be defended here.

There are several problems with the simplest view. First, feature structure formalisms normally make no provision for quantifiers or other higher-order operators. Second, there seems to be no way of defining a lambda operator in these formalisms. See Moore 1989 for a discussion of the difficulties this entails in the case of type-raising. Third (and generalizing over the last two points), a semantic representation language must be extremely powerful if it is to represent the range of the possible meanings in natural language. The needs of a feature formalism which excludes semantics are much more modest, which allows the formalism to retain better computational properties. If we tried to represent semantics directly in feature structures, we would be left with a representation scheme so rich that its computational properties could not be attractive. It is worth noting that several research strategies deal with this difficulty either by avoiding it—concentrating on applications which do not exercise the richer expressive capabilities of natural language, or by ignoring it—employing nonlogical semantic representation schemes whose expressive and computational properties are not understood exactly. The latter tack suffers in many obvious ways, but particularly when it comes to attempting inference on any but an *ad hoc* basis. The former tack—seeking applications with reduced semantics demands—is certainly defensible, but very difficult to maintain because semantic complexity can arise in unexpected places. A particular problem with this strategy arises in interface applications, where the "flexibility" of natural language interfaces is adduced as a justification for their development. Flexibility invariably requires rich expressive means. Of course, the present proposal can claim nothing more than to restrict complexity to one aspect of linguistic analysis and to prevent it from infecting others unnecessarily.[1]

[1] I have deliberately omitted monotonicity considerations as an argument for the metalinguistic interpretation of feature-based semantics, since, if I am right that what is at issue is a question of the interpretation of the feature-descriptions, then monotonic processing must be possible, even on the direct interpretation view. What then appear to be violations of monotonicity (e.g., negation does not merely restrict interpretation) can then always be explained away as confusions about which parts of a description describe the same object (in the case of negation, the scope of negation must not be unified with its result).

2.1 What is Ambiguity?

In this section I would like to review a conceptual puzzle which Nerbonne 1992a poses. The problem is solved neatly by the metalinguistic view, but it remains puzzling on the direct denotation view. This concerns the distinction between vagueness (or generality) and ambiguity, and will be introduced by an example. The noun *bank* is ambiguous between the '$-home' and 'riverside' meanings in a way that *glove* is not, even though *glove* can refer only to left-hand gloves or right-hand gloves—quite distinct categories of objects. This distinction is crucial in quantification and anaphora. Thus *three gloves* ignores the distinction between left- and right-hand gloves in a way that *three banks* may not (this cannot refer to a pair of financial institutions and a riverside); similarly *Sam bought a bank and Bob sold one* cannot describe a situation involving a financial institution on the one hand and a riverside on the other; i.e., it allows only a pair of the four possible combinations of the two meanings. The vague (or general) *glove* is less restrictive; any of the four possible combinations of meanings can be at play in the following example:[2]

Sam lost a glove and Bob found one

We need to distinguish ambiguity and vagueness (or generality). If semantics distinguishes two levels, a level of meaning specification and a level of meaning (or denotation), then ambiguity is a disjunction (or underspecification) at the first level, while vagueness (or generality) is disjunction (or underspecification) at the second. If semantics is compositional or "direct" (and thus involves no essential level of logical form) then the notion of disjunction leads to the absurd consequences, i.e., that there is no distinction between vagueness and ambiguity. This is the crux of a final argument for the metalinguistic view of semantic features.

Let us examine the metalinguistic view in more detail. If we allow that semantics makes metalinguistic specifications, then it specifies that 'bank' has the meaning

[PRED: {$-HOME,RIVERSIDE}]

, using metalinguistic disjunction. Under the (quite reasonable) assumption that there is no single object-language predicate denoted by this disjunction, we can explain the quantificational and anaphoric facts. On this view the only predicates described by the above specification are just $-HOME and RIVERSIDE, so the description must be of exactly one of those. *Glove*, on the other hand, is unambiguous, even if its semantics may be equivalent to an object-language disjunction: [PRED: GLOVE], where the metalinguistic

[2] Cf. Zwicky and Sadock 1975 for a discussion of anaphora as a test of ambiguity.

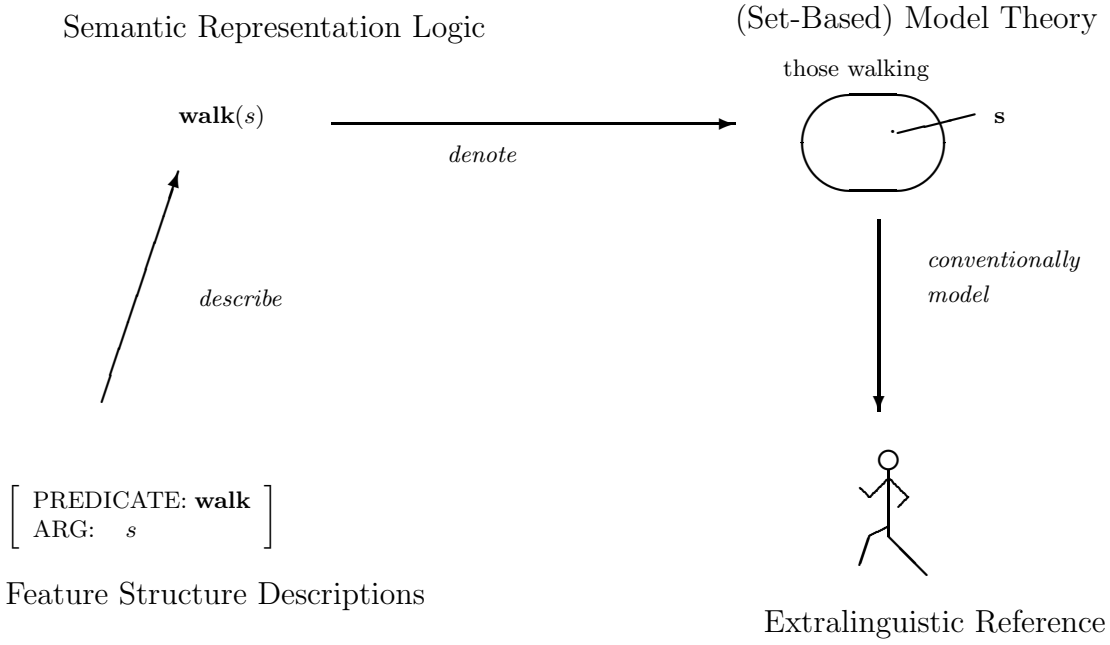


Figure 2: The relation of feature-based semantics to semantic representation logic, model theory, and the extralinguistic objects of discourse. A simpler, but ultimately unsatisfactory view, would eliminate the semantic representation logic.

'GLOVE' denotes the object language predicate **glove**, which may satisfy the object-language equivalence below:

$$\textbf{glove}(x) \text{ iff } \textbf{left-glove}(x) \vee \textbf{right-glove}(x)$$

I would like to anticipate two objections to the argument presented here. The first objection is that lexical ambiguity is simply the existence of two distinct lexical items with the same pronunciation, etc., and the second objection is that the apparent puzzle above dissolves if finer-grained semantic theories are employed. We examine these in turn.

The standard representation of lexical ambiguity is that of distinct lexical items, but this solution essentially denies purely semantic ambiguity, reducing it to an unanalyzed notion of 'distinct lexical item', and it fails to generalize to the view of the lexicon as a disjunction of words, common in feature-based theories (cf. HPSG, Pollard and Sag 1987 p.147). Under the view of the lexicon as a disjunction of words (or word descriptions), the postulation of lexical items distinct only in semantics reduces to the postulation of a single item with disjunctive semantics, since:

$$(p_1 \wedge ... \wedge p_n \wedge q) \vee (p_1 \wedge ... \wedge p_n \wedge r) \Leftrightarrow (p_1 \wedge ... \wedge p_n \wedge (q \vee r))$$

The postulation of distinct lexical items does not rescue the direct interpretation tack since it is equivalent—via the factoring above—to the postulation of a single item with disjunctive semantics. Thus this attempted solution boils down again to whether semantics makes metalinguistic specifications, or whether it denotes directly.

The solution offered here is consistent with the view that multiple lexical entries are involved, but it immediately suggests a more perspicuous representation (in analogy to the right-hand side of the biconditional); it differs only in insisting that there be two levels of semantics: a level which constrains semantic representations, and the level of semantic representations themselves. Lexical ambiguity involves underspecification at the first level; lexical vagueness (or generality) is underspecification at the second.

We turn now to the second objection, that this puzzle of ambiguity vs. vagueness (or generality) dissolves if finer-grained semantic theories are adopted. There are two points to be made against this objection, first, that semantic grain does not seem to be the only issue, and second, that, if it is indeed the issue, then a very rough grain would seem to be appropriate (at least for quantification). We discuss each now in more detail. We can illustrate the objection arising from the consideration of semantic grain in the context of the example above. The objection need only suppose that the predicates specified as semantics for *bank*, i.e., $-HOME and RIVERSIDE denote different entities—perhaps different predicate atoms, as they might in situation semantics (Barwise and Perry 1983). On this interpretation, we might then suppose that the only entities which satisfy the disjunctive description {$-HOME, RIVERSIDE} are the relevant atoms. And thereafter, the treatment would be just as in the case of the metalinguistic interpretation. —We can respond to this objection in the following way. The difficulty here is in the supposition that the only entities satisfying the disjunctive description will be the relevant atomic predicates. This amounts to a prohibition of disjunctive properties which could be quite difficult to maintain (*He must be citizen or resident, Any citizen or resident may apply*). In this sense the issue is not just the possibility of using a finer-grained semantics, but also the nonexistence of disjunctive properties. And even if a tenable semantics could be sketched without disjunctive properties, this line of objection would seem committed to disallowing disjunctive entities across the board. I would expect to be able to raise the ambiguity vs. vagueness (generality) issue wherever disjunctive entities were allowed.

A second rebuttal to the objection from semantic grain would allow the relevance of the objection, but ask how fine-grained an appropriate semantics is. Quantification facts argue that the predicates denoted by common noun phrases are distinguished only by their extensions. This is, e.g., implied by the conservativity postulate in generalized quantifier theory (Barwise and Cooper 1981). Thus it will not do simply to point to logics in which there may be a relation of material equivalence, but no relation of logical equivalence between the left and right sides in the *glove* equation above. This is not motivated except as a means of solving the ambiguity/vagueness puzzle since natural language quantification is insensitive to distinctions finer than material equivalence. But if extensions were the level of semantic sensitivity, and if we were interpreting directly, then we should expect a quantification such as *three banks* NOT to count financial institutions and riversides indiscriminately. This concludes our discussion of the objections.

Given a level of semantic representation together with a metalinguistic level at which constraints on semantic representation are expressed, a notion of semantic ambiguity as opposed to semantic underspecification may be characterized. This is further justification for the view that semantic processing involves the manipulation of semantic representation—which in turn further justifies the postulation of this level of representation in addition to the level at which meanings are directly modeled. The more general claim

on which this argument hinges is that the characterization of ambiguity in a representation L_1 is always with respect to a second representational system L_2. The scheme proposed here distinguishes two levels of semantics and is thus capable of characterizing ambiguity; systems with a single level are not.

The view adopted here is that there is a semantic representation language, distinct from the feature formalism, and that feature structures may be profitably employed as a formal metalanguage for this semantic representation scheme. A lengthy illustration is provided in Section 4 below. Under this view feature structure expressions describe expressions in a logic designed for semantic representation in natural language. Figure 2 provides a schematic view of the proposed view of the relation between feature structure descriptions and their ultimate semantic referents, the objects spoken of.

The following sections illustrate in detail how a semantics conceived along these lines functions. They show how scope ambiguity can be characterized in this sort of system—without reference to auxiliary notions such as quantifier stores or quasi-logical forms. The ability to characterize such meanings should constitute a further challenge to those who wish to interpret feature descriptions of semantics directly.

3 An Illustration

We illustrate the view that semantics involves the accumulation of constraints expressed about a semantic representation language, by applying the view to a well-known semantic representation language, the language of generalized quantifiers; second, by developing its metalanguage within the feature description language used in HPSG; and third by demonstrating its application to problems of predicate disambiguation and scope.

3.1 Logic—A Generalized Quantifier Language

To illustrate the approach we shall first need a target semantic representation language. Here we deliberately use a popular semantic representation scheme—a version of the language of generalized quantifiers; this is a kind of *lingua franca* among theoretically oriented computational linguists. We emphasize that the overall scheme—that of employing feature structure descriptions as a formalized metalanguage for a semantic representation logic—is general, and could easily be applied to other logics, e.g., first order logic, higher-order or intensional logics, discourse representation structures, or a language of situation theory.. It could even be applied to nonlogical representations, but that would be harder to motivate.

A BNF for a Language of Generalized Quantifiers—LGQ

$\langle \text{var} \rangle \Rightarrow x_0, x_1, \ldots$

$\langle \text{const} \rangle \Rightarrow c_0, c_1, \ldots$

$\langle \text{pred} \rangle \Rightarrow P_0, P_1, \ldots$

$\langle \text{term} \rangle \Rightarrow \langle \text{var} \rangle \mid \langle \text{const} \rangle$

$\langle \text{atomic-wff} \rangle \Rightarrow (\langle \text{pred} \rangle \langle \text{term} \rangle *)$

$\langle \text{conn} \rangle \Rightarrow \wedge, \vee, \leftrightarrow$

$\langle \text{det} \rangle \Rightarrow \forall, \exists, MOST, \ldots$

$\langle \text{g-quant} \rangle \Rightarrow \langle \text{det} \rangle \langle \text{var} \rangle \langle \text{wff} \rangle$

$\langle \text{wff} \rangle \Rightarrow \langle \text{atomic-wff} \rangle$
$\mid (\langle \text{wff} \rangle \langle \text{conn} \rangle \langle \text{wff} \rangle)$
$\mid (\neg \langle \text{wff} \rangle)$
$\mid (\langle \text{wff} \rangle \rightarrow \langle \text{wff} \rangle)$
$\mid (\langle \text{g-quant} \rangle \langle \text{wff} \rangle)$

There is one detail about this syntax definition which may seem peculiar to non-computational linguists. The definition foresees quantified formulas not in the Barwise-Cooper notation (cf. Barwise and Cooper 1981) where the quantified formula normally had λ-terms in the restrictor and scope, but rather in the notation more frequently used in computational linguistics, in which these positions are filled by open formulas. The formula below highlights the difference:

Barwise and Cooper Notation	**PresentNotation**
$\forall x(\lambda y.\mathbf{man}(y), \lambda z.\mathbf{mortal}(z))$	$(\forall x\ \mathbf{man}(x), \mathbf{mortal}(x))$

This is generally preferred in order to keep the complexity of formulas (the number of λ's) at a minimum. Cf. Moore 1981 for an early use; and Dalrymple et al. 1991, pp.414-17 for model-theoretic definitions of the alternative forms.

We turn now to the description of expressions in this language using a typed feature description language of the sort common in grammar processing.

3.2 Metalogic—AVM Specifications for LGQ

In this section we employ typed feature logic (often referred to as a ATTRIBUTE-VALUE MATRICES or AVMs) to provide a set of type definitions for expressions in the logical language just presented. We

shall not present the typed feature logic formally, relying on (Carpenter 1992) for definitions. The initial specifications are limited to very vanilla-flavored uses of the feature description scheme, and we shall warn when more particular assumptions are made.

$$\langle \text{var} \rangle \Rightarrow x_0, x_1, \ldots \qquad \begin{bmatrix} \textit{var} \\ \text{INDEX} \end{bmatrix}$$

$$\langle \text{atomic-wff} \rangle \Rightarrow (\langle \text{pred} \rangle \langle \text{term} \rangle *) \qquad \begin{bmatrix} \textit{atomic-wff} \\ \text{PRED } [\textit{pred}] \\ \text{FIRST } [\textit{term}] \\ \vdots \\ \text{N-TH } [\textit{term}] \end{bmatrix}$$

$$\langle \text{g-quant} \rangle \Rightarrow \langle \text{det} \rangle \langle \text{var} \rangle \langle \text{wff} \rangle \qquad \begin{bmatrix} \textit{gen-quant} \\ \text{DET } [\textit{det}] \\ \text{VAR } [\textit{var}] \\ \text{REST } [\textit{wff}] \end{bmatrix}$$

$$\langle \text{wff} \rangle \Rightarrow \ \ldots \mid (\langle \text{wff} \rangle \langle \text{conn} \rangle \langle \text{wff} \rangle) \qquad \begin{bmatrix} \textit{connective-wff} \\ \text{CONN } [\textit{conn}] \\ \text{WFF1 } [\textit{wff}] \\ \text{WFF2 } [\textit{wff}] \end{bmatrix}$$

$$\ldots \mid (\neg \langle \text{wff} \rangle) \qquad \begin{bmatrix} \textit{negation} \\ \text{SCOPE-WFF: } [\textit{wff}] \end{bmatrix}$$

$$\ldots \mid (\langle \text{wff} \rangle \ \rightarrow \ \langle \text{wff} \rangle) \qquad \begin{bmatrix} \textit{implication} \\ \text{ANTE } [\textit{wff}] \\ \text{CONSEQ } [\textit{wff}] \end{bmatrix}$$

$$\ldots \mid (\langle \text{g-quant} \rangle \langle \text{wff} \rangle) \qquad \begin{bmatrix} \textit{q-wff} \\ \text{QUANT } [\textit{gen-quant}] \\ \text{SCOPE } [\textit{wff}] \end{bmatrix}$$

It is easy to note the absence of several expressions types from the set of feature-structure types defined; for example, terms, predicate and individual constants have not been defined here. That is because we rely on TYPE information for distinctions which are not realized in one or more distinct attributes. Figure 3 illustrates the type hierarchy we assume.

3.3 Use of Semantic Descriptions—Examples

Some simple examples of the kinds of metalinguistic specifications allowed are illustrated in Figure 4. These would be compiled in ways suggested by Figure 1; i.e., we imagine that construction principles (or grammatical rules) may have a semantic correlate which constrains the semantic representation which is the meaning of the construction. Of course, as we noted above, there is no reason that only syntax should have the privilege of constraining meaning.

The use of feature structures as a metalinguistic level of semantic representation allows greater freedom in semantic explanations. Cf. Nerbonne 1992a for illustration.

4 A Richer Constraint Language

The use of feature descriptions above (ignoring the use of types) is of the fairly simple sort common to all feature systems, including, e.g., PATR-II (cf. Shieber et al. 1983). In general, this was conceived as a theory of linguistic categories (Gazdar et al. 1987), which complemented a "backbone" of context-free phrase-structure rules. Initially, this division of labor—feature-structures on the one hand and CF rules on the other—appeared justified by the need for recursion in the latter, but not in the former. But the treatment of long-distance dependence appears to necessitate the use of some recursion even in the feature-structure component (cf. Kaplan and Zaenen 1988 and Kaplan and Maxwell 1988). This has justified the attempt to develop feature structure description languages which can describe both the context-free and the categorial components of language—and which therefore foresee a fairly liberal use of recursion.

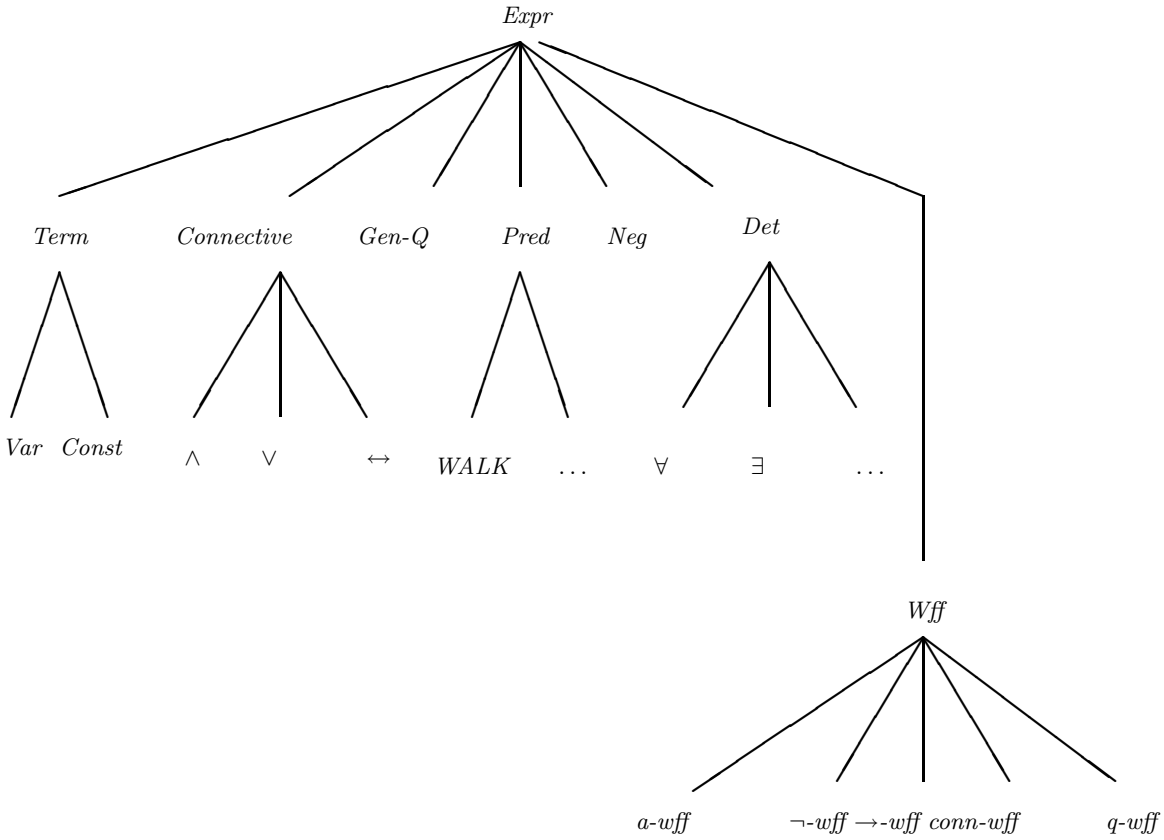


Figure 3: A type hierarchy for the domain of expressions in the language of generalized quantifiers as defined in the text. Carpenter, 1992 provides the theory of typed feature structures within which a hierarchy such as this functions.

In continuing below our investigations in constraint-based semantics, we shall employ the language of HPSG (Pollard and Sag 1987) because HPSG, more than any other current school of computational linguistics, promotes the richer view of feature structure description. Indeed, HPSG may be seen as the formulation of the feature-based work followed to its logical conclusion: here, feature structures provide not merely a theory of linguistic categories, but rather an alternative foundation for all linguistic theory.

A precise foundation for the more elaborate HPSG language, which we shall sketch and employ informally, is provided in Carpenter 1992. Briefly, it consists of a set of terms described in attribute-value matrices of the sort employed above, which, however, are subject to CONSTRAINTS expressed in a richer language. The constraint language allows TYPING, i.e., restrictions on appropriateness of attributes and values; RELATIONS between feature terms; as well as references to SETS. The motivation and technical development for set terms is provided in Pollard and Moshier 1990. One aspect of Pollard and Moshier's treatment is particularly significant below: a description set of n element descriptions can never hold of sets of $n + m$ elements, but it can hold of sets of $n - m$ individuals—this is possible where some of the element descriptions are compatible. (Rounds 1988 provides alternative developments of the notion 'set' in feature logic.)

5 Scope Underspecification

A fundamental problem in computational semantics is the support of DISAMBIGUATION, i.e., the resolution of semantic ambiguity and underspecification. Applications of feature-based techniques for predicate disambiguation may be found in Moens et al. 1989 and Nerbonne 1992b. Scope disambiguation is a more difficult problem both because the relevant factors are less well understood, and because the underspecified information is more complex. In this section we investigate the use of the HPSG feature-description language as a means of describing semantic representations which are underspecified with respect to scope.

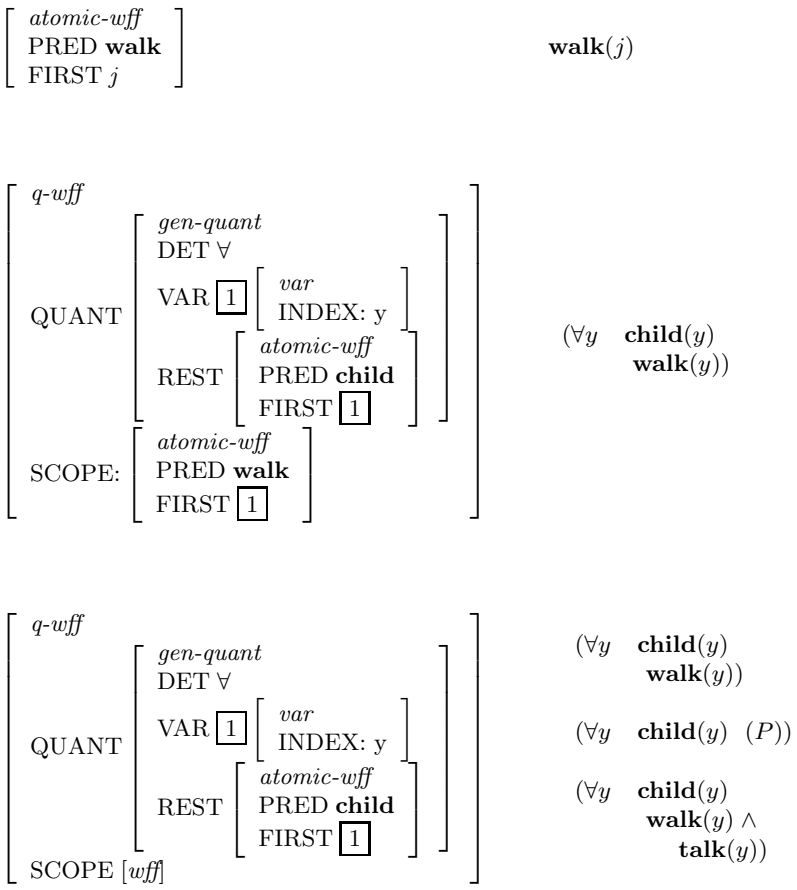


Figure 4: Feature structure descriptions used as metalinguistic descriptors of expressions in LGQ. Note the use of underspecification in the last example.

To provide the flavor of the system proposed, we first provide the (relatively simple) specifications in feature-description language for two common metalogical tasks—the definition of free variable, and that of closed formula. Each of these is employed in the following treatment of scope, as is a third, that of formula without vacuous binding, whose definition may also be found below.

5.1 Metalogical Task—Definition of Free Variables

We add to each type a feature FREE-VARS, which is constrained by type definitions. The feature description specification mimics the usual recursive definition (cf. Ebbinghaus et al. 1978, p.30). A variable (occurrence) is free in atomic formulas, and free in boolean combinations under the obvious conditions, and finally free in quantified formulas where it is free in a component and not bound by the quantifier itself. The following definitions in typed feature logic formalize this:

$$\left[\begin{array}{l} \textit{atomic-wff} \\ \text{PRED} \\ \text{FIRST ARG}_1 \\ \vdots \\ \text{NTH ARG}_n \\ \text{FREE-VARS } \boxed{1} \end{array}\right] \qquad \text{Condition: } x \in \boxed{1} \text{ iff } x = \text{ARG}_i,\ 1 \leq i \leq n \text{ and } \textit{var}(x)$$

$$\left[\begin{array}{l} \textit{connective-wff} \\ \text{CONN} \\ \text{WFF1 } \boxed{2} \\ \text{WFF2 } \boxed{3} \\ \text{FREE-VARS } \boxed{1} \end{array}\right] \qquad \begin{array}{l} \text{Condition: } x \in \boxed{1} \text{ iff} \\ \qquad x \in \boxed{2}|\text{FREE-VARS or } x \in \boxed{3}|\text{FREE-VARS} \end{array}$$

(And similarly for conditional and negated wffs.)

$$\left[\begin{array}{l} \textit{gen-quant} \\ \text{DET} \\ \text{VAR } \boxed{2} \\ \text{REST} \\ \text{FREE-VARS } \boxed{1} \end{array}\right] \qquad \text{Condition: } x \in \boxed{1} \text{ iff } x \in \text{REST}|\text{FREE-VARS and } x \neq \boxed{2}$$

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT} \\ \text{SCOPE} \\ \text{FREE-VARS } \boxed{1} \end{array}\right] \qquad \begin{array}{l} \text{Condition: } x \in \boxed{1} \text{ iff } x \in \text{QUANT}|\text{FREE-VARS or} \\ \qquad (x \neq \text{QUANT}|\text{VAR and } x \in \text{SCOPE}|\text{FREE-VARS}) \end{array}$$

Given the definition of free variables, the definition of closed formula is straightforward:

$$\textit{closed-wff} = \left[\begin{array}{l} \textit{wff} \\ \text{FREE-VARS } \emptyset \end{array}\right]$$

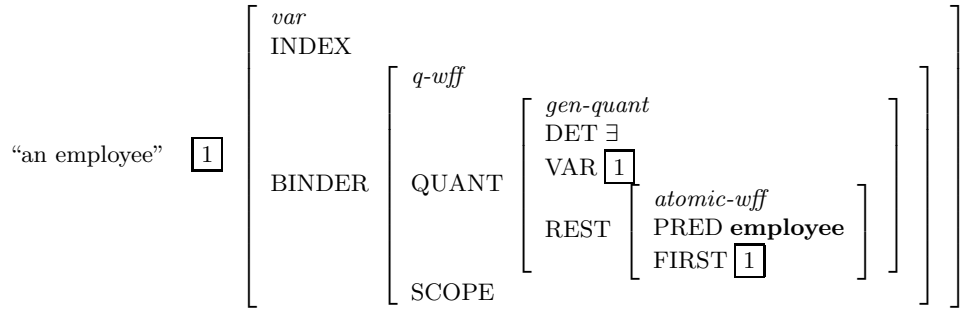
5.2 Overspecifying Logical Forms

Note that the means of defining free variables (above), depends on adding superfluous information to metalogical specifications. The FREE-VARS attribute provides information which is useful, but calculable from other information present. It provides for overspecified feature structures—overspecified vis-à-vis LGQ. For example, an atomic formula with no arguments but a nonempty FREE-VARS attribute would be overspecified (and its specification would fail to satisfy the associated constraint). We exploit this possibility in what follows.

An interesting opportunity arises when we consider slightly less obvious descriptions of logical formulas. Suppose, for example, that the type *var* contains a field, not only for the variable name, but also for its scope, i.e., the quantified formula whose quantifier binds the variable:

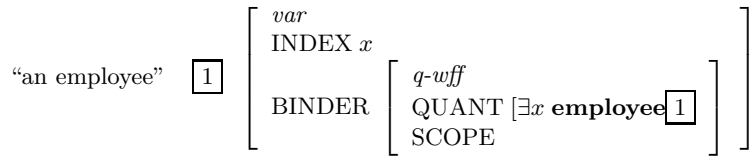
$$\langle \text{var} \rangle \Rightarrow x_0, x_1, \ldots \qquad \left[\begin{array}{l} \textit{var} \\ \text{INDEX} \\ \text{BINDER } [\textit{q-wff}] \end{array}\right]$$

Since the scope of a bound variable is unique, this is a well-defined feature (and something similar is common in implementations of theorem provers). It can now be exploited to represent scopally underspecified structures. For example, we can now represent formulas in which argument positions are bound to variable arguments, which in turn point to unique quantified formulas, whose scopes, however, needn't be specified. An example of a term in such a structure would be the following:



The point of employing a representation such as this is, of course, that it encodes all of the information associated with the binding of a quantified term to a particular argument position, but none of the information associated with the scope the quantifier takes. It can thus serve as part of a representation for underspecified scopes.

Before demonstrating how this sort of representation might be employed, it will be useful to introduce an abbreviatory convention. As the feature representation of the quantified formulas above amply demonstrates, the standard logic representations are considerably more concise than the representations proposed here (in feature descriptions). Because of this, we shall abbreviate lengthy feature descriptions with the logic formulas they describe, wherever this can be done with no danger of confusion. Thus for the structure above we shall write:



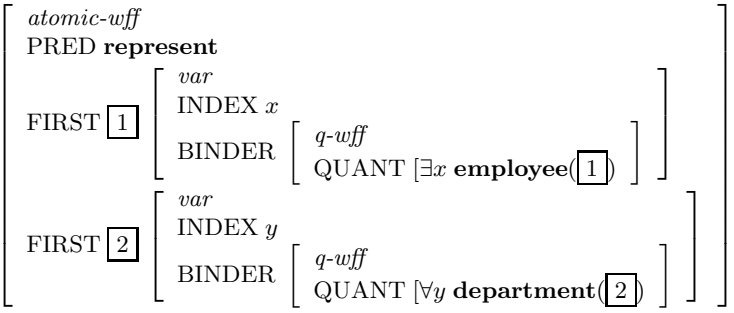
Where there is no possibility of confusion, we use the (more compact) logical notation to abbreviate AVM's describing the logic. There are two equivalent feature descriptions here, where the first hides somewhat more information than the second (which corresponds more closely to the implementation). The first normally suffices, but some details (e.g., about the quasi-logical forms discussed in the following section) are more easily understood on the basis of the second.

5.3 Quasi-Logical Form (QLF)

In this section we employ the representations proposed in the last to provide a level of representation in which scope is unspecified. It represents a brief diversion in the overall develop in that the structures proposed here will not find use later on. Instead, they will be used as examples of scopally underspecified structures of popular, but *sui generis* type—neither logical formulas nor descriptions of them. The point of this section is to contrast this development with the one advocated in the main body of the paper.

First, we assume that the quantified formulas in the different VAR-BINDER positions have themselves unspecified scopes, in order to obtain a structure whose information content resembles the CLE's "quasi-logical forms" (Alshawi and others July 1989), in that it shows which quantifiers bind which argument positions, but it says nothing about the relative scopes of the quantifiers. It is crucial to note here that the structure below does NOT actually describe any quantified formulas at all—it merely describes an atomic formula whose argument positions are filled by variables. The parallel to Schubert and Pelletier's and Alshawi et al.'s structures is genuine—we have defined nonlogical structures from which genuine semantics are readily derived, just as they did.

(1)



Although this sort of representation is generally known under the CLE term "quasi-logical form" (hence: QLF), it is for all intents and purposes just Schubert and Pelletier 1982's "ambiguous logical syntax", or Hobbs and Shieber 1987's "wff's with complex terms", or Fenstad et al. 1987's "unscoped situation schemata". The idea of deriving (during parsing) unscoped representations from which scoped formulas could be algorithmically derived is standard (and good) practice among theoretically minded computational semanticists.

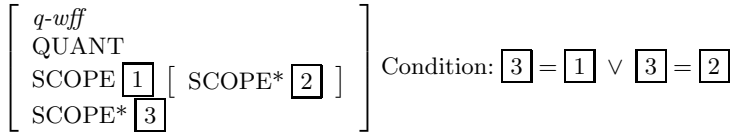
But like all good practice, it invites theoretical questions as to WHAT it assumes and WHY it is effective. We have nothing to say definitively on the latter (agreeing with the widespread prejudice that it must have to do with the feasibility of ignoring or at least postponing scope disambiguation). The former question is particularly interesting here, because feature-based semantics can provide a general account of the nature of QLF: it simply represents one way of partially specifying semantic constraints. But QLF is theoretically suspect because it is an *ad hoc* representation. Here we can take a step further: given our general scheme of allowing the statement of constraints over logical forms, there is no need for a distinguished level of QLF; i.e., we can encode the information in QLF in the semantic metalanguage under development here.

Before showing how this is possible, it is worth noting that while the representation in (1) includes all the information in QLF, still it does not DESCRIBE the disambiguated scopings—instead, it stands in an algorithmic relation to them. That is, there is an algorithm which maps from representations such as these to scope disambiguations. The representation (1) thus only makes sense within a system with a level of QLF. What is distinctive about the feature-based systems under discussion here is that one can ELIMINATE the level of QLF, even while preserving all the information it contains. This is possible if the opportunity for underspecification in feature structures is exploited properly.

(1) therefore only suggests how QLF might be specified; we would prefer, however, to eliminate QLF in favor of variously (under)specified semantic feature structures. We seek a single representation which (in the sense of feature logics) subsumes various scope possibilities. These possibilities should be obtained by further specifications of feature values.

5.4 Underspecified Scopes

The information in a quasi-logical form (above) is NOT that a particular quantified-wff has a particular scope, but rather EITHER that a particular quantified-wff has a particular scope OR that its scope has a particular scope, OR that its scope's scope...etc. This motivates the addition of an apparently redundant feature:



We require that the feature SCOPE* is defined wherever SCOPE is (on *q-wff*). Intuitively, SCOPE* is the kernel, or nuclear scope of a complex formula; it functions as a regular path specification of the sort used by Kaplan and Maxwell 1988. Furthermore, *quantified-wff* is subject to a type restriction that SCOPES* is either identical to the value of SCOPE or to the value of SCOPE* in the scope.

This feature appears redundant, but it is only redundant when formulas are fully specified. It allows us to represent quasi-logical forms in a fashion that genuinely subsumes scopally unambiguous formulas. But to show this we likewise need to characterize the quantifiers involved, so that we add a second apparently ambiguous feature, with a similar type restriction:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \boxed{1} \\ \text{SCOPE } \left[\begin{array}{l} \text{QUANT* } \boxed{2} \end{array}\right] \\ \text{QUANT* } \boxed{3} \end{array}\right] \text{ Condition: } \boxed{3} = \{* \boxed{1} *\} \cup \boxed{2}$$

Starred braces denote set delimiters. We again require that QUANT* is defined wherever QUANT is (on *q-wff*). Intuitively, QUANT* is the set of quantifiers involved in a formula. The condition in this case is again recursive (and no longer expressible as a regular path expression), viz., that QUANT* be the union of the SCOPE's QUANT* together with the singleton set of the local QUANT, so that QUANT* is the set of quantifiers along the path to an atomic scope. Given these definitions, we can now specify a feature structure which subsumes the various scope ambiguities and which allows that the level of "quasi-logical form" be eliminated. We examine the feature structure description below, intended as a representation of the following sentence (where scope is underspecified):

An employee represents each department

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT* } \{* \exists x \, \mathbf{employee}(x), \forall y \, \mathbf{department}(y) *\} \\ \text{SCOPE* } [\mathbf{represent}(x, y)] \end{array}\right]$$

This structure describes formulas whose dominant operator is one of the two quantifiers in QUANT*, whose scope feature is unspecified ($\top$), and for which the atomic-wff $\mathbf{represent}(x, y)$ occurs on the path SCOPE*. The structure may be further specified e.g., by setting the scope feature (to be one of the two quantifiers involved).

Given the definitions above, this feature description can be true of exactly two formulas. Proof: QUANT must have a value, and QUANT* must contain it (def.). Choose one of the two possibilities, the second will be the (parallel) second solution. Given choice of quantifier, SCOPE|QUANT* must have other quantifier, and SCOPE|SCOPE* must be identical to SCOPE* (since otherwise type clash). Thus above reduces to:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \exists x \, \mathbf{employee}(x), \\ \text{SCOPE } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT* } \{* \forall y \, \mathbf{department}(y) *\} \\ \text{SCOPE* } \mathbf{represent}(x, y) \end{array}\right] \end{array}\right]$$

We examine the value of SCOPE. It must define QUANT, which must therefore be equal to the element in the singleton QUANT* (since the latter had to contain the QUANT value). Thus SCOPE|SCOPE cannot be anything with a QUANT value—and therefore nothing with a SCOPE or SCOPE* value. But SCOPE|SCOPE* must be either SCOPE|SCOPE or
SCOPE|SCOPE|SCOPE*. Since the latter cannot be defined, it must be the former. Thus:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \exists x \, \mathbf{employee}(x), \\ \text{SCOPE } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \forall y \, \mathbf{department}(y) \\ \text{SCOPE } \mathbf{represent}(x, y) \end{array}\right] \end{array}\right]$$

The second reading involves choosing the second quantifier first, but is otherwise parallel. □

This indicates that the required subsumption relations indeed hold of this formula (and that no further logic formulas are described by this AVM). Thus scope disambiguation can be characterized at this level. It is clear that the characterization above crucially involved Pollard & Moshier set descriptions, which can never hold of sets with more elements than the set description has. For those who would prefer other notions of sets, the required restrictions can be obtained in interesting ways. For this we need two further notions: first, a restriction to closed formulas, as defined above.Second, we need a way to ban vacuous quantification, as provided in Appendix A.

The scope ambiguity problem is classic, and the Schubert and Pelletier and Alshawi et al. solution to it is generally recognized to be the best available, and certainly one of the most practical. But it rests on the theoretically questionable foundation of "quasi-logical form". The point of demonstrating that one can obtain the same predictions as the "quasi-logical form" theory in a theoretically motivated fashion—through the use of feature structures as a formalized metalanguage for logic—is to show the capability of the feature-based approach as a foundation for semantics and disambiguation.

5.5 Formulas with no Vacuous Binding—*nvb-wff*

In examining the range of reasonable realizations of scopally underspecified formulas, it is helpful to exclude from consideration formulas with vacuous binding. Here we provide the needed definitions.

atomic-wff $\sqsubseteq$ *nvb-expr*

$$\begin{bmatrix} \textit{connective-wff} \\ \text{CONN} \\ \text{WFF1}\ \boxed{1} \\ \text{WFF1}\ \boxed{2} \\ \text{FREE-VARS} \end{bmatrix}$$

for $x \ni$ *connective-wff*(x), *nvb-expr*(x) iff *nvb-expr*($\boxed{1}$) and *nvb-expr*($\boxed{2}$)
(And similarly for conditional and negated wffs.)

$$\begin{bmatrix} \textit{gen-quant} \\ \text{DET} \\ \text{VAR}\ \boxed{1} \\ \text{REST}\ \boxed{2} \\ \text{FREE-VARS} \end{bmatrix}$$

for $x \ni$ *gen-quant*(x), *nvb-expr*(x) iff *nvb-expr*($\boxed{2}$) and $\boxed{1} \in \boxed{2}$|FREE-VARS
(And similarly for quantified wffs.)

6 Nested Scopes

In this section we extend the approach to account for "nested" quantification, as this is treated in Keller 1988. Our target analysis is the sentence below, which ought to have either of the two analyses following:

An expert in every field attended the meeting

Targets: fully specified scopes

$$\begin{bmatrix} \textit{q-wff} \\ \text{QUANT} \begin{bmatrix} \textit{gen-quant} \\ \text{DET}\ \exists \\ \text{VAR}\ x \\ \text{REST} \begin{bmatrix} \textit{q-wff} \\ \text{QUANT}\ \forall y\ \textbf{field}(y) \\ \text{SCOPE}\ \textbf{expert-in}(x,y) \end{bmatrix} \end{bmatrix} \\ \text{SCOPE} \begin{bmatrix} \textit{atomic-wff} \\ \textbf{attended-meeting}(x) \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \textit{q-wff} \\ \text{QUANT}\ \forall y\ \textbf{field}(y) \\ \text{SCOPE} \begin{bmatrix} \textit{q-wff} \\ \text{QUANT}\ \exists x\ \textbf{expert-in}(x,y) \\ \text{SCOPE} \begin{bmatrix} \textit{atomic-wff} \\ \textbf{attended-meeting}(x) \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

In order to accommodate this within the feature-based scheme introduced above, we first need to define a third apparently redundant feature, REST*, transitive derivative of REST (the restrictor of a quantifier).

$$\begin{bmatrix} \textit{gen-quant} \\ \text{REST}\ \boxed{1}\ [\ \text{SCOPE*}\ \boxed{2}\ ] \\ \text{REST*}\ \boxed{3} \end{bmatrix} \text{Condition: } \boxed{3} = \boxed{1} \vee \boxed{3} = \boxed{2}$$

Just as in the previous cases, we require that REST* be defined wherever REST is (on *gen-quant*). Intuitively, REST* is the kernel, or nuclear scope of a complex quantifier.

Second, we require a slight revision of the definition of QUANT*, the transitive derivative of SCOPE and QUANT. (The revision will not affect the representation and not require significant modifications in the proof in the text.)

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \boxed{1} \ \left[\ \text{REST } [\text{QUANT* } \boxed{2}]\ \right] \\ \text{SCOPE } \left[\ \text{QUANT* } \boxed{3}\ \right] \\ \text{QUANT* } \boxed{4} \end{array}\right] \text{ Condition: } \boxed{4} = \{* \boxed{1} *\} \cup \boxed{2} \cup \boxed{3}$$

Intuitively, QUANT* is still the set of quantifiers involved in a formula, now just defined more comprehensively. N.b., the treatment of nonnested quantifiers still works (but the formulation above now needs a restriction to closed formulas).

We illustrate the treatment with an application to the sentence:

An expert in every field attended the meeting

This receives the following scopally underspecified representation:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT* } \{* \forall y \ \mathbf{field}(y), \left[\begin{array}{l} \text{DET } \exists \\ \text{VAR } x \\ \text{REST* } \mathbf{expert\text{-}in}(x,y) \end{array}\right] *\} \\ \text{SCOPE* } \left[\begin{array}{l} \textit{atomic-wff} \\ \mathbf{attended\text{-}meeting}(x) \end{array}\right] \end{array}\right]$$

There are exactly two solutions. Proof: QUANT must be filled; choose one from QUANT*.

1. $\forall y$ **field**(y). Then the remaining cannot come from QUANT|REST|QUANT*, since this would leave no way of binding the x in the SCOPE*. (Note the implicit appeal to the definition of closed formula, provided in Section 5.1 above.) It is therefore reducible to:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \forall y \ \mathbf{field}(y) \\ \text{SCOPE } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT* } \{* \left[\begin{array}{l} \text{DET } \exists \\ \text{VAR } x \\ \text{REST* } \mathbf{expert\text{-}in}(x,y) \end{array}\right] *\} \\ \text{SCOPE* } \left[\begin{array}{l} \textit{atomic-wff} \\ \mathbf{attended\text{-}meeting}(x) \end{array}\right] \end{array}\right] \end{array}\right]$$

   Since SCOPE|QUANT must be filled, the quantifier in SCOPE|QUANT* must fill it. Since there are no further quantifiers involved, there can be no SCOPE|SCOPE|SCOPE*, which leaves SCOPE|SCOPE to be filled by SCOPE|SCOPE*. This yields the fully specified formula:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \forall y \ \mathbf{field}(y) \\ \text{SCOPE } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \exists x \ \mathbf{expert\text{-}in}(x,y) \\ \text{SCOPE } \left[\begin{array}{l} \textit{atomic-wff} \\ \mathbf{attended\text{-}meeting}(x) \end{array}\right] \end{array}\right] \end{array}\right]$$

2. $\left[\begin{array}{l} \text{DET } \exists \\ \text{VAR } x \\ \text{REST* } \mathbf{expert\text{-}in}(x,y) \end{array}\right]$ Once this is put into QUANT, it is clear that the remaining quantifier, $\forall y$ **field**(y), cannot be in SCOPE|QUANT* (since this would leave the variable y in QUANT|REST* unbound). The remaining quantifier therefore must be in QUANT|REST|QUANT*, yielding:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } \left[\begin{array}{l} \text{DET } \exists \\ \text{VAR } x \\ \text{REST* } \mathbf{expert\text{-}in}(x,y) \\ \text{REST|QUANT* } \{* \forall y \ \mathbf{field}(y) *\} \end{array}\right] \\ \text{SCOPE* } \left[\begin{array}{l} \textit{atomic-wff} \\ \mathbf{attended\text{-}meeting}(x) \end{array}\right] \end{array}\right]$$

Since there's no further QUANT* specification here, the SCOPE cannot involve a quantifier, so that SCOPE* must be SCOPE. QUANT|REST, on the other hand, has a QUANT* value, and therefore MUST have a QUANT value, which moreover, must be the element in the singleton QUANT|REST|QUANT. But QUANT|REST*—an atomic-wff—is incompatible with QUANT|REST, meaning it must therefore be QUANT|REST|SCOPE*. This gives us:

$$\left[\begin{array}{ll} \textit{q-wff} & \\ \text{QUANT} & \left[\begin{array}{ll} \text{DET} & \exists \\ \text{VAR} & x \\ \text{REST} & \left[\begin{array}{ll} \text{QUANT} & \forall y\ \textbf{field}(y) \\ \text{SCOPE*} & \textbf{expert-in}(x,y) \end{array}\right] \end{array}\right] \\ \text{SCOPE} & \left[\begin{array}{l} \textit{atomic-wff} \\ \textbf{attended-meeting}(x) \end{array}\right] \end{array}\right]$$

But now, since QUANT|REST|QUANT* contains no further quantifiers, QUANT|REST|SCOPE cannot be of type *q-wff*, so that QUANT|REST|SCOPE* must be QUANT|REST|SCOPE. This is just the second reading:

$$\left[\begin{array}{ll} \textit{q-wff} & \\ \text{QUANT} & \left[\begin{array}{ll} \text{DET} & \exists \\ \text{VAR} & x \\ \text{REST} & \left[\begin{array}{ll} \text{QUANT} & \forall y\ \textbf{field}(y) \\ \text{SCOPE} & \textbf{expert-in}(x,y) \end{array}\right] \end{array}\right] \\ \text{SCOPE} & \left[\begin{array}{l} \textit{atomic-wff} \\ \textbf{attended-meeting}(x) \end{array}\right] \end{array}\right]$$

Thus exactly the two desired readings are characterized. □

Several refinements are potentially interesting: first, we may generalize to account for scope ambiguity involving arbitrary operators (including, e.g., negation and modal operators) by creating a supertype of *quantified-wff*, say *operator-wff* and defining OPERATOR* and SCOPE* in analogy to QUANT* and SCOPE* above, respectively; second, we need to demonstrate how one could translate into the underspecified representation (cf. Nerbonne et al. 1993); third, it would be most interesting to identify, represent and exploit disambiguating factors; and fourth, it would be worthwhile exploring the use of underspecified representations which are not actually subjected to disambiguation (cf. Poesio 1991, Reyle 1992 for proposals for semantic representation schemes in which scopes are unspecified).

7 Conclusions and Prospectus

The purpose of the present paper has been the investigation of the use of feature structures for the purpose of semantics processing. We have argued that "semantic" feature structures are best conceived as constraints on logical form. This is equivalent to the view that the feature description language functions as a formalized metalanguage for an arbitrary semantic representation language. This view of the use of semantic feature structures recommends itself by the relatively few assumptions it makes about the nature of the semantic representation language itself—in particular, it need not be limited to the expressive power of the feature logic metalanguage. A further advantage for this view is the opportunity it affords for the characterization of semantic ambiguity. We illustrated the proposal using an application to the language of generalized quantifiers, and an extended excursion into the relatively rich HPSG formalism demonstrated a feature-based approach to the characterization of scopally underspecified formulas which eschews the level of "quasi-logical forms".

Finally, we must note that the program of genuinely integrating syntax and semantics is not realized completely here, since some semantic processing, namely genuinely semantic inference, must be separate from the construction of semantic forms. This problem exists as well e.g., in Montague's program, where syntax must first be mapped into logic for any inference to occur. The feature-based theories may have an advantage here in that it may be possible to define (weak) semantic consequence relations directly on the feature structures, but this is a topic for later work.

Referenties

Alshawi, H., et al. July 1989. Research Programme in Natural Language Processing. Final report. SRI Cambridge Research Centre: Alvey Project No. ALV/PRJ/IKBS/105.

Barwise, Jon, and Robin Cooper. 1981. Generalized Quantifiers and Natural Language. *Linguistics and Philosophy* 4(2):159–219.

Barwise, Jon, and John Perry. 1983. *Situations and Attitudes*. Cambridge: MIT Press.

Carpenter, Bob. 1992. *The Logic of Typed Feature Structures*. Tracts in Theoretical Computer Science, No. 32. Cambridge: Cambridge University Press.

Carpenter, Bob, Carl J. Pollard, and Alex Franz. 1991. The Specification and Implementation of Constraint-Based Unification Grammar. In *Proc. of the Second International Workshop on Parsing Technology*. Cancun.

Dalrymple, Mary, Stuart M. Shieber, and Fernando C.N.Periera. 1991. Ellipsis and Higher-Order Unification. *Linguistics and Philosophy* 14(4):399–452.

Ebbinghaus, Heinz-Dieter, Jörg Flum, and Wolfgang Thomas. 1978. *Einführung in die Mathematische Logik*. Darmstadt: Wissenschaftliche Buchgesellschaft.

Fenstad, Jens Erik, Per-Kristian Halvorsen, Tore Langholm, and Johan van Benthem. 1987. *Situations, Language, and Logic*. Dordrecht: Reidel.

Gazdar, Gerald, Geoffrey K. Pullum, Robert Carpenter, Ewan Klein, Thomas Hukari, and Robert D. Levine. 1987. Category Structures. Technical Report CSLI-87-102. CSLI.

Hobbs, Jerry R., and Stuart M. Shieber. 1987. An Algorithm for Generating Quantifier Scopings. *Computational Linguistics* 13(1-2).

Kaplan, Ronald, and John Maxwell. 1988. An Algorithm for Functional Uncertainty. In *Proc. of Coling 1988*, 297–302. Budapest.

Kaplan, Ronald, and Annie Zaenen. 1988. Long-Distance Dependencies, Constituent Structure, and Functional Uncertainty. In *Alternative Conceptions of Phrase Structure*, ed. Mark Baltin and Anthony Kroch. xxx–yyy. Chicago: University of Chicago Press.

Kasper, Robert T., and William C. Rounds. 1986. A Logical Semantics for Feature Structures. In *Proc. of the 24th Annual Meeting of the Association for Computational Linguistics*, 257–266. Columbia University.

Keller, William R. 1988. Nested Cooper Storage: The Proper Treatment of Quantification in Ordinary Noun Phrases. In *Natural Language Parsing and Lingusitic Theories*, ed. Uwe Reyle and Christian Rohrer. 432–447. Dordrecht: Reidel.

Moens, Marc, Jo Calder, Ewan lein, Mike Reape, and Henk Zeevat. 1989. Expressing Generalizations in Unification-Based Grammar Formalisms. In *Proc. of the 4th Meeting of the European Chapter of the Association for Computational Linguistics*, 174–181.

Moore, Robert C. 1981. Problems in Logical Form. In *Proc. of the 19th Annual Meeting of the Association for Computational Linguistics*, 117–124.

Moore, Robert C. 1989. Unification-Based Semantic Interpretation. In *Proc. of the 27th Annual Meeting of the Association for Computational Linguistics*, 33–41.

Nerbonne, John. 1992a. Constraint-Based Semantics. In *Proc. of the 8th Amsterdam Colloquium*, ed. Paul Dekker and Martin Stokhof, 425–444. Institute for Logic, Language and Computation. also DFKI RR-92-18.

Nerbonne, John. 1992b. Natural Language Disambiguation and Taxonomic Reasoning. In *DFKI Workshop on Taxonomic Reasoning (DFKI D-92-08)*, ed. Jochen Heinsohn and Bernhard Hollunder, 40–47. Saarbrücken. DFKI.

Nerbonne, John, Joachim Laubsch, Abdel Kader Diagne, and Stephan Oepen. 1993. Software for Applied Semantics. In *Proc. of Pacific Asia Conference on Formal and Computational Linguistics*, ed. Chu-Ren Huang, Claire Hsun hui Chang, Keh jiann Chen, and Cheng-Hui Liu, 35–56. Taipei. Academica Sinica. Also available as DFKI Research Report RR-92-55.

Poesio, Massimo. 1991. Relational Semantics and Scope Disambiguation. In *Proc. of the Second Conference on Situation Theory and Its Applications*, ed. Jon Barwise, Jean Mark Gawron, and Gordon Plotkin. Stanford University: CSLI Lecture Notes.

Pollard, Carl, and Drew Moshier. 1990. Unifying Partial Descriptions of Sets. In *Information, Language and Cognition*, ed. P.Hanson. Vancouver Studies in Cognitive Science, Vol. 1, 167–184. Vancouver: University of British Columbia Press.

Pollard, Carl, and Ivan Sag. 1987. *Information-Based Syntax and Semantics, Vol.I.* Stanford: CSLI.

Reyle, Uwe. 1992. Dealing with Ambiguities by Underspecification: A First-Order Calculus for Unscoped Representations. In *Proc. of the 8th Amsterdam Colloquium*, ed. Paul Dekker and Jaap van der Does, xxx–yyy. Amsterdam.

Rounds, William C. 1988. Set Values for Unification-Based Grammar Formalisms and Logic Programming. Technical Report CSLI-88-129. Stanford University: Center for the Study of Language and Information.

Schubert, Lenhart K., and Francis Jeffrey Pelletier. 1982. From English to Logic: Context-Free Computation of Conventional Logic Translations. *Journal of the Association for Computational Linguistics* 165–176.

Shieber, Stuart. 1986. *An Introduction to Unification-Based Approaches to Grammar*. Stanford University: Center for the Study of Language and Information.

Shieber, Stuart, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and Mabry Tyson. 1983. The Formalism and Implementation of PATR-II. In *Research on Interactive Acquisition and Use of Knowledge*. Menlo Park, Calif.: Artificial Intelligence Center, SRI International.

Smolka, Gert. 1988. A Feature Logic with Subsorts. Technical Report 33. WT LILOG–IBM Germany.

Zwicky, Arnold M., and Jerrold Sadock. 1975. Ambiguity Tests and How to Fail Them. In *Syntax and Semantics, Vol. IV*, ed. John Kimball. 1–36. New York: Academic Press.