Constraint-Based Semantics

John Nerbonne*
Alfa-informatica
Rijksuniversiteit Groningen
Oude Kijk in 't Jatstraat 41
NL 9700 AS Groningen, The Netherlands
nerbonne@let.rug.nl

Abstract

Montague's famous characterization of the homomorphic relation between syntax and semantics naturally gives way in computational applications to CONSTRAINT-BASED formulations. This was originally motivated by the close harmony it provides with syntax, which is universally processed in a constraint-based fashion. Employing the same processing discipline in syntax and semantics allows that their processing (and indeed other processing) can be as tightly coupled as one wishes—indeed, there needn't be any fundamental distinction between them at all. In this paper, we point out several advantages of the constraint-based view of semantics processing over standard views. These include (i) the opportunity to incorporate nonsyntactic constraints on semantics, such as those arising from phonology and context; (ii) the opportunity to formulate principles which generalize over syntax and semantics, such as those found in HEAD-DRIVEN PHRASE STRUCTURE GRAMMAR; (iii) a characterization of semantic ambiguity, which in turn provides a framework in which to describe disambiguation, and (iv) the opportunity to underspecify meanings in a way difficult to reconcile with other views. The last point is illustrated with an application to scope ambiguity in which a scheme is developed which underspecifies scope but eschews auxiliary levels of logical form.

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

*Thanks to Bob Carpenter, Kris Halvorsen, Bob Kasper, Uli Krieger, Joachim Laubsch and Carl Pollard for discussion of the ideas presented here.

Constraint-Based Semantics

Contents

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. We assume throughout this paper a familiarity with feature-based linguistic description; Shieber 1986 is the fundamental introduction.

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.

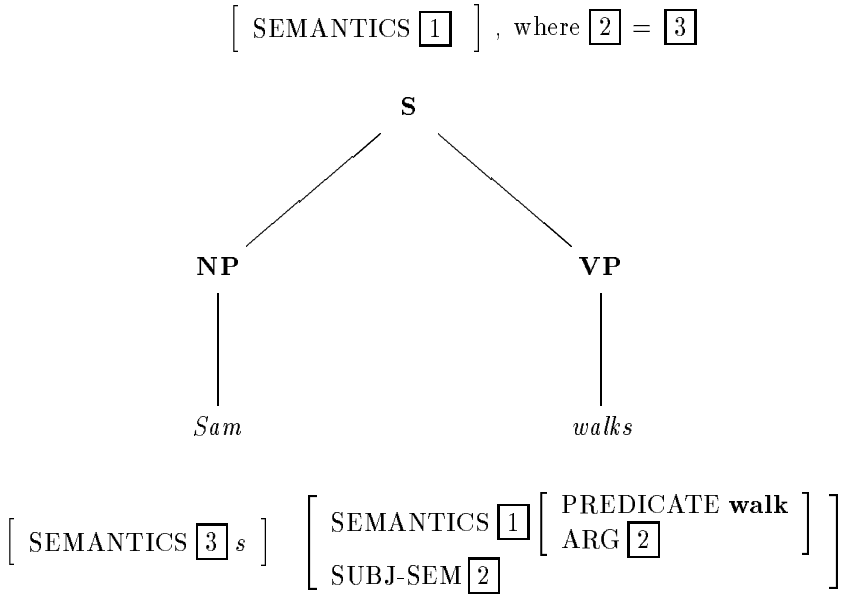


Figure 1: A sketch of the semantic derivation of *Sam walks*, **walk** (s) as this would proceed using unification. Unification applies to syntactic and semantic representations alike, eliminating the need to compute these in distinct processes, and unification is employed to bind variables to argument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

This paper adduces several advantages of a constraint-based view of the syntax/semantics relation over standard views, including (i) the opportunity to incorporate nonsyntactic constraints on semantics, such as those arising from phonology or context; (ii) the opportunity to formulate principles which generalize over syntax and semantics, such as those found in HEAD-DRIVEN PHRASE STRUCTURE GRAMMAR (Pollard and Sag 1987). We then argue that feature-based semantics ought to be understood as specifying constraints on logical forms, not model structures. The virtues of viewing semantics processing as manipulation of logical form descriptions includes not only a characterization of semantic ambiguity, which in turn provides a framework in which to describe disambiguation, but also the opportunity to underspecify meanings in a way difficult to reconcile with other views. The latter point is illustrated with an application to the notorious scope ambiguity problem.

2 Constraint-Based Semantics

There are several advantages of the unification-based view of the syntax/semantics interface over the more familiar (Montogovian) view of this interface, which is characterized by a homomorphism from syntax into semantics. The constraint-based view sees the interface as characterized by a set of constraints to which nonsyntactic information may contribute, including phonological and pragmatic information. We consider these in Sections 2.2 and 2.3 below, and Fenstad et al. 1987, pp.12-17 discusses the distinction between the constraint-based and the homomorphic views of the relation between syntax and semantics from processing and other more general points of view.

We first examine the opportunity for expressing syntax/semantics generalizations which the constraint-based view provides, and then turn to the semantics of sentence accent and that of deixis as examples which indicate that there should be no homomorphism from

syntax (narrowly understood) to semantics. In its place we argue that a constraint-based view is preferable.

2.1 Syntax/Semantics Generalizations

The constraint-based of the syntax/semantics relationship allows syntactic and semantic information to be bundled in complex, but useful ways. This contrasts with the Montgagovian view which sees them as domain and range of a homomorphism—objects of potentially very different kinds. The opportunity to bundle information in complex ways allows the very simple statements of syntax/semantics relationships that make HPSG attractive (e.g. the SUBCATEGORIZATION PRINCIPLE (Pollard and Sag 1987, p.71) ultimately responsible for the semantics computation shown in Figure 1).

To see this, let us provide a more general set of specifications for the example in Figure 1. We may begin by noting that verb phrases not only allow reference to their subject semantics, they also may impose restrictions on their subjects (e.g., that they agree with the head of the VP). This leads us to a specification in which not only SUBJ-SEM is specified on the VP, but more generally SUBJECT, which in turn specifies potentially both syntax and semantics. But we note that SUBJECT is only one of several potential complements, each of which could be specified in this fashion, leading us to a view of the VP as follows:

$$\left[\begin{array}{l} \text{SEMANTICS } \boxed{1}\left[\begin{array}{l}\text{PREDICATE } \textbf{walk}\\ \text{ARG } \boxed{2}\end{array}\right] \\ \text{COMPLEMENTS } \langle \left[\begin{array}{l}\text{SEMANTICS } \boxed{2}\\ \text{SYN|AGR } 3s\end{array}\right] \rangle \end{array}\right]$$

But the sense of this sort of specification is that it contributes to the definition of well-formed phrases, in this case by specifying the complements with which a head may combine (as in categorial grammar). This is provided by a restriction on the complement-head phrasal type, the SUBCATEGORIZATION PRINCIPLE:

$$\left[\begin{array}{l} \text{SEMANTICS } \boxed{1} \\ \text{DAUGHTERS } \left[\begin{array}{l} \text{HEAD-DTR } \left[\begin{array}{l}\text{SEMANTICS } \boxed{1}\\ \text{COMPLEMENTS } \langle \boxed{2}, \ldots \rangle\end{array}\right] \\ \text{COMPLEMENT-DTR } \boxed{2}\end{array}\right]\end{array}\right]$$

Taken together (and assuming an NP specification of the sort provided in Figure 1), these specifications guarantee that the correct semantics (for a large subset of examples) are provided. What is striking here is that no special semantics principle of combination is required—the subcategorization principle effects a unification of a complement daughter with a head's complement specifications. But since semantics are among these, the principle has as a consequence that the argument position corresponding to the complement semantics is bound.

Before concluding this section, we should note that my presentation of the subcategorization principle abstracts away from an essential part, in that the actual subcategorization principle also accounts for the COMPLEMENTS specifications of phrases as these are derived from the phrase's components—in much the same fashion as categorial grammar. But this is merely additional, not conflicting information. The interested reader may consult Pollard and Sag 1987, p.87. Pollard 1989 investigates a more speculative interweaving of syntax and semantics.

2.2 Semantics/Phonology

Focus-sensitive adverbs such as *only* depend for their interpretation on phonological sentence accent, and not only on syntactic structure. (I should like to emphasize that the point is NOT

that syntax plays no role in the interpretation of focus, since it surely does; e.g., when it is noted that phonological sentence accent must be realized within a focused constituent—i.e., within a syntactically defined unit, the definitions depend on syntax. The point is merely that the phonology, too, requires semantic interpretation.)

A treatment of the semantics of focus which provides correct semantics for sentences such those in (1) is found in Rooth 1985:

(1) a. Tom only introduced BILL to Sue.
b. Tom only introduced Bill to SUE.

(where uppercase is used to designate accented words.) Rooth's treatment assigns a single semantics—independently of accent—to the VP *introduce Bill to Sue* in both (1a) and (1b), i.e.

$$\lambda\, x.\mathbf{introduce}(x, b, s)$$

The semantics of the particle *only*, used here adverbially, applies to this property to derive another whose intuitive content is just the property of individuals who have no "contextually" relevant property other than that denoted by the VP:

$$\lambda\, y(\forall\, P[P(y) \wedge C(P) \;\rightarrow\; P = \lambda\, x.\mathbf{introduce}(x, b, s)]$$

Thus Rooth's treatment relies on a specification of the ALTERNATIVES to a given assertion to provide a semantics for focus.

The restriction to contextually relevant properties solves an important technical difficulty which arises when one tries to specify that an individual has only a single property: we cannot simply ignore properties of self-identity, etc. $\lambda\, x\, .x = x$. Failing this restriction, the specification in question *only introduce BILL to Sue* could only be satisfied in counterintuitive models—those with a single element, denoted by each of *Tom, Bill* and *Sue*.

The specification of 'C', the "contextually relevant" properties, is delimited by focus (sentence accent). For example, the properties required in the case of (1a) are defined as follows:

$$C \;=\; \{P|\exists\, x\; P \;=\; \lambda\, y.\mathbf{introduce}(y, x, s)\}$$

This stands today as one of relatively few successful treatments of focus phenomena.[1] But one can ask whether it would not be intuitively preferable to derive the meanings required not via functions whose sole input is the meanings of phrases, but rather via constraints based on both phrase meanings and phonological properties. In that case, we should postulate that the VP *introduce BILL to Sue* has following semantic and phonological properties:

$$\left[\begin{array}{l} \textit{VP} \\ \text{SEMANTICS } \mathbf{introduce}(x, y, \mathbf{sue}) \\ \text{PHONOLOGY|FOCUS|SEMANTICS|} \left[\begin{array}{l} \text{PLACEHOLDER } y \\ \text{CONTENT } \mathbf{bill} \end{array}\right] \end{array}\right]$$

In this we ascribe semantic properties to phonological entities, but this is plausible given their ultimate source in sentence accent. In addition, we require a lexical specification of *only*:

[1] An interesting recent alternative is provided by Krifka 1991, building on von Stechow 1989. As will become apparent, my point here is not to argue for Rooth's analysis, but rather to show how a similar analysis fits rather nicely into a constraint-based view of semantics. So I will not try to justify using Rooth's work rather than Krifka's except to say that Rooth's is earlier and better known.

$$\left[\begin{array}{l} \textit{Adv} \\ \text{SEMANTICS } \forall\,\boxed{1}(\boxed{3} \leftrightarrow \boxed{1} = \boxed{2}) \\ \text{MOD} \left[\begin{array}{l} \text{SEMANTICS } \boxed{3} \\ \text{PHONOLOGY|FOCUS|SEMANTICS|} \left[\begin{array}{l} \text{PLACEHOLDER } \boxed{1} \\ \text{CONTENT } \boxed{2} \end{array}\right] \end{array}\right] \end{array}\right]$$

Given two general principles about the interpretation of adjuncts, viz. (i) that the semantics of adjunct head combinations is derived from the adjunct, and (ii) that the adjunct's MOD specifications must unify with the head, these specifications yield the result that the VP *only introduce BILL to Sue* means:

$$\forall\, y(\mathbf{introduce}(x, y, \mathbf{sue}) \leftrightarrow y = \mathbf{bill})$$

(We do not specify the VP meaning as a λ-abstract since we do not rely on function application to derive meanings.) This is equivalent in all relevant respects to Rooth's analysis except in that it identifies the source of focus as phonological rather than syntactic.

2.3 Semantics/Context

In this section we would like to show that a constraint-based approach is preferable to a homomorphic one because it provides a natural modus for the expression of constraints arising from context.

A number of contemporary theories of meaning are radically noncompositional with respect to the role played by discourse context. In these theories semantics is NOT effected via a homomorphism from syntax alone, but rather via a two-place mapping from syntax and context (or pragmatics). Situation semantics (Barwise and Perry 1983, Gawron and Peters 1990) is merely a stricter codification of prevalent views in this respect. Noncompositionality is most evident in the analysis of nonquantified noun phrases in Gawron and Peters 1990, which has its roots in the analysis of "singular noun phrases" in Barwise and Perry 1983. The semantic contribution a nonquantified NP makes is simply an individual to which the NP refers—but the determination of WHICH individual depends on the context in which the NP is uttered, and not only on the meanings of the NP's syntactic constituents.

This may be illustrated using deictic expressions. Gawron and Peters might analyze an utterance of *That dog walks* as having the logical form $\mathbf{walk}(x)$, and as importing the contextual restrictions that x denotes only if it holds both that $\mathbf{dog}(x)$ and $\mathrm{REFREL}(that, x)$. The latter condition obtains when an utterance of *that* is used to refer to the object assigned to x. Of course, if either of these conditions is not met, then the utterance of the expression *that dog* fails to denote, and no assertion is made.

For our programmatic argument the important point is not merely that the context of utterance plays a role in determining the semantic contribution of a phrase (surely a trivial property of the semantics of deictics), but also that this contextual dependence is not a part of the semantic contribution made by deictics, i.e., that there are apparently no semantic contexts which use or modify the contextual dependency in anyway. The deictics are "rigid designators", whose denotation exhausts their semantic contribution.

Gawron and Peters introduce RESTRICTED PARAMETERS as a technical aid in explicating the context dependence of nonquantificational noun phrases. Restricted parameters are logical terms of the form '$(x|\,\phi)$' for x an individual variable, ϕ a condition (state-of-affairs), which denote the individual assigned to x in case ϕ holds, and which otherwise fail to denote. This looks like a perfectly compositional (partial) definition, except that the conditions involved may involve conditions on context, e.g., 'REFREL'. But we may highlight the importance of constraint-based view more effectively by providing a formulation in the

feature specification language employed here throughout. A candidate for the right sort of representation (for our example *That dog walks*) is provided:

$$\left[\begin{array}{l} \textit{sign} \\ \text{SEMANTICS} \left[\begin{array}{l} \text{CONTENT } \mathbf{walk}(x) \\ \text{CONTEXT REFREL}(\textit{that}, x) \wedge \mathbf{dog}(x) \end{array}\right] \end{array}\right]$$

According to the Gawron and Peters view, the usual recursive definition of SEMANTICS (based on DAUGHTER|SEMANTICS and SYNTAX) would give way to simultaneous recursive definitions of SEMANTICS|CONTENT and SEMANTICS|CONTEXT (based on DAUGHTER|SEMANTICS|CONTENT, DAUGHTER|SEMANTICS|CONTEXT and SYNTAX).

There are of course several responses available to adherents of compositionality. The most frequent response is to reinterpret meanings so that contextual dependency is inherent in them. (We have in mind the work on context in Kaplan 1979.) This saves compositionality at the expense of weakening it, perhaps completely, since context is a very broad concept.

Important for our present purposes is the fact that there are theories of semantic representation according to which semantics is noncompositional, and that the constraint-based view accommodates these comfortably.

3 The Nature of Constraint-Based Semantics

The usefulness of the feature-based view of semantics temporarily postpones the foundational question of the nature of the semantic representations being employed. The postponement is only temporary, however, since an answer to the question is crucial to several issues in the practical use of these representations:

- 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 Johnson 1988; Reape 1991 and Blackburn 1991 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. The present proposal does NOT claim to banish computational complexity from linguistic formalisms—but it does shield the feature formalism from the complexity (incompleteness) inherent in semantic representation schemes.

Finally, I would like to demonstrate a conceptual problem which the metalinguistic view solves neatly, but which is quite puzzling on the direct denotation view. This concerns the distinction between vagueness 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); similary *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

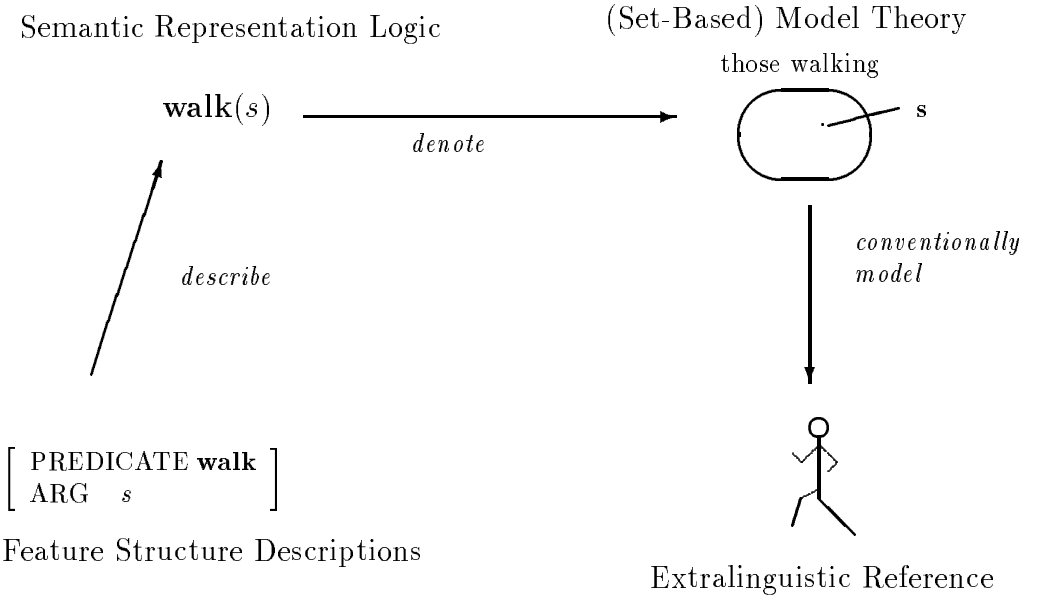


Figure 2: The relation of feature-based semantics to semantic representation logic, model theory, and the extralinguistic objects of discourse. A simpler, but ultimately unsatisfactory view, would eliminate the semantic representation logic.

combinations of the two meanings. The vague (or underspecified *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

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. *Glove*, on the other hand, is unambiguous, even if its semantics may be equivalent to an object-language disjunction: [PRED **glove**], where the object-language disjunction holds:

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

The standard solution here is to postulate distinct lexical items for ambiguity, but this solution essentially denies purely semantic ambiguity, reducing it to lexical ambiguity, and it fails to generalize to the view of the lexicon as a disjunction of words, common in the 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:

$$(p_1 \wedge \ldots \wedge p_n \wedge q) \vee (p_1 \wedge \ldots \wedge p_n \wedge r) \leftrightarrow (p_1 \wedge \ldots \wedge p_n \wedge (q \vee r))$$

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 is underspecification at the second.[3]

Thus, 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.

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

[3] While there is not space here to anticipate all the reactions to this argument, I would like to note that the quantification facts show that this is not an issue of semantic grain, in the sense in which this is debated, e.g., in situation semantics and possible worlds semantics. Cf. Barwise and Perry 1983, Barwise 1989. 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 above. This is so because natural language quantification is insensitive to distinctions finer than material equivalence.

The Section 4 illustrates in more detail how a semantics conceived along these lines functions.

4 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 scope.

4.1 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

$\langle \text{var} \rangle ::= x_0 \mid x_1 \mid \ldots$

$\langle \text{const} \rangle ::= c_0 \mid c_1 \mid \ldots$

$\langle \text{pred} \rangle ::= P_0 \mid P_1 \mid \ldots$

$\langle \text{term} \rangle ::= \langle \text{var} \rangle \mid \langle \text{const} \rangle$

$\langle \text{atomic-wff} \rangle ::= (\langle \text{pred} \rangle \langle \text{term} \rangle^*)$

$\langle \text{conn} \rangle ::= \wedge \mid \vee \mid \leftrightarrow$

$\langle \text{det} \rangle ::= \forall \mid \exists \mid MOST \mid \ldots$

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

$$\begin{aligned} \langle \text{wff} \rangle ::= \; & \langle \text{atomic-wff} \rangle \\ & \mid (\langle \text{wff} \rangle \langle \text{conn} \rangle \langle \text{wff} \rangle) \\ & \mid (\neg \langle \text{wff} \rangle) \\ & \mid (\langle \text{wff} \rangle \rightarrow \langle \text{wff} \rangle) \\ & \mid (\langle \text{g-quant} \rangle \langle \text{wff} \rangle) \end{aligned}$$

There is one detail about this syntax definition which may seem peculiar to non-computational semanticists. 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 visual 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.

4.2 Metalogic—AVM Specifications for LGQ

In this section we employ typed feature logic (often referred to as 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 (Section 5).

$\langle \text{var} \rangle ::= x_0 \mid x_1 \mid \ldots$

$$\begin{bmatrix} \textit{var} \\ \text{INDEX} \end{bmatrix}$$

$\langle \text{atomic-wff} \rangle ::= (\langle \text{pred} \rangle \langle \text{term} \rangle^*)$

$$\begin{bmatrix} \textit{atomic-wff} \\ \text{PRED } [\textit{pred}] \\ \text{FIRST } [\textit{term}] \\ \vdots \\ \text{N-TH } [\textit{term}] \end{bmatrix}$$

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

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

$\langle \text{wff} \rangle ::= \ldots \mid (\langle \text{wff} \rangle \langle \text{conn} \rangle \langle \text{wff} \rangle)$

$$\begin{bmatrix} \textit{connective-wff} \\ \text{CONN } [\textit{conn}] \\ \text{WFF1 } [\textit{wff}] \\ \text{WFF2 } [\textit{wff}] \end{bmatrix}$$

$\ldots \mid (\neg \langle \text{wff} \rangle)$

$$\begin{bmatrix} \textit{negation} \\ \text{SCOPE-WFF } [\textit{wff}] \end{bmatrix}$$

$\ldots \mid (\langle \text{wff} \rangle \rightarrow \langle \text{wff} \rangle)$

$$\begin{bmatrix} \textit{implication} \\ \text{ANTE } [\textit{wff}] \\ \text{CONSEQ } [\textit{wff}] \end{bmatrix}$$

$\ldots \mid (\langle \text{g-quant} \rangle \langle \text{wff} \rangle)$

$$\begin{bmatrix} \textit{q-wff} \\ \text{QUANT } [\textit{gen-quant}] \\ \text{SCOPE } [\textit{wff}] \end{bmatrix}$$

We shall make use of TYPE predicates, e.g., $\textit{var}(x)$, which holds iff x is of type *var*. It is easy to note the absence of several expression 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. Of course, we can express the type hierarchy in the language of typed feature descriptions—and in this way obtain a specification fully equivalent to the BNF. For example, in Carpenter 1992 we can specify that

$$\textit{term}(x) \leftrightarrow \textit{var}(x) \vee \textit{const}(x)$$

For the purposes of this paper, we may rely on the informal presentation in Figure 3.

4.3 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 con-

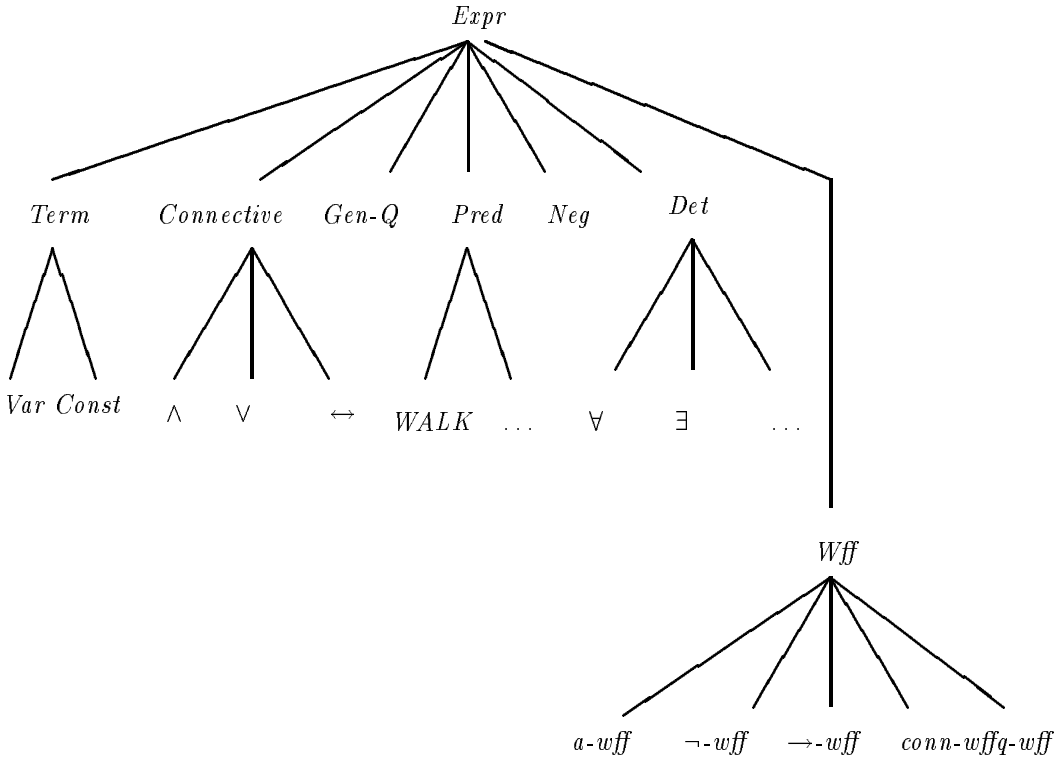


Figure 3: A type hierarchy for the domain of expressions in the language of generalized quantifiers as defined in the text. Carpenter 1992 provides the theory of typed feature structures within which a hierarchy such as this functions.

struction 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. This may be illustrated with respect to the representation of argument positions in atomic formulas and their specifications in the current scheme. The semantic representation language LGQ uses order-coding to represent which arguments are bound to which argument positions, while the metalanguage (feature structure descriptions) identifies this using features (FIRST, etc.). The first bit of freedom we might exercise concerns the identification of argument positions. Nothing would stand in the way of using more contentful-sounding role names to pick these out. For example, we might alter the feature specifications in such a way as to allow the following:

$$\begin{bmatrix} \textit{atomic-wff} \\ \text{PRED } \mathbf{send} \\ \text{SOURCE } x \\ \text{THEME } y \\ \text{GOAL } z \end{bmatrix}$$

In this case the simplest generalization would appear to be that allowing any name to

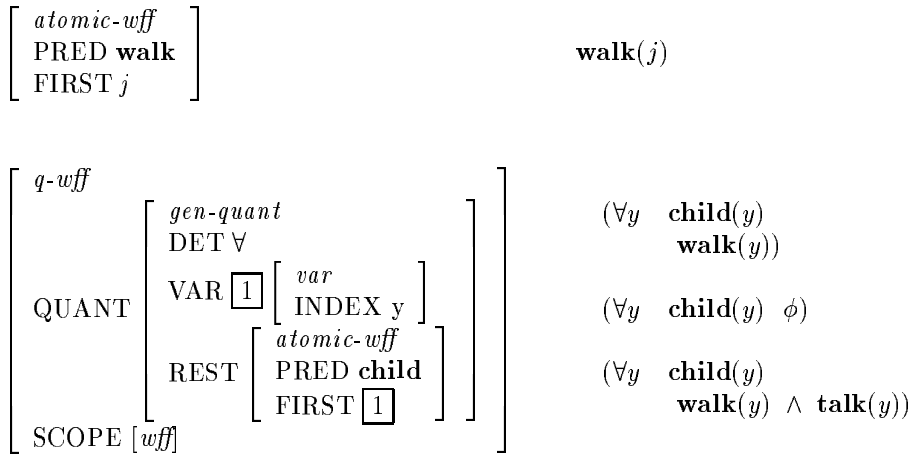


Figure 4: Feature structure descriptions used as metalinguistic descriptors of expressions in LGQ. Note the use of underspecification in the last example. The underspecified description is compatible with many, including semantically contradictory formulas.

designate a semantic role. We would again represent roles as features, postponing any more detailed representation until it is motivated:

$$\langle\text{atomic-wff}\rangle ::= (\langle\text{pred}\rangle\langle\text{pair}\rangle^*) \qquad \begin{bmatrix} \textit{atomic-wff} \\ \text{PRED } [\textit{pred}] \\ \text{ROLE}_1\ [\textit{term}] \\ \vdots \\ \text{ROLE}_n\ [\textit{term}] \end{bmatrix}$$

Although we will not make use of this representation below, it illustrates a degree of freedom allowed by the present semantic representation scheme but absent from simpler ones. It would moreover appear to suffice for all the syntactic purposes for which so-called thematic roles are deployed (cf. Jackendoff 1972, 29-46; Roberts 1991; Dowty 1991; Wechsler 1991).

On the other hand, there are purely semantic grounds for preferring keyword-coding to order-coding as a way of identifying argument positions. Two such reasons often adduced are (i) that the ORDER of arguments in relations is never used semantically, and thus that every alternative ordering leads to a perfectly equivalent logic, so that the keyword-coding suffices; and (ii) that the use of keyword coding allows us to make sense of anadic predication (i.e., using the same predicate with a variable number of arguments). On the latter point, see Creary and Pollard 1985.

5 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 necessitates the use of some recursion even in the feature-structure component (cf. Kaplan and Zaenen 1988 and Kaplan and Maxwell 1988). In continuing be-

low our investigations in constraint-based semantics, we shall employ the language of HPSG (Pollard and Sag 1987) because HPSG proceeds from the richer view of feature structure description languages. 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. We rely on Carpenter 1992 for formal development of the HPSG feature description language, consisting of feature terms to which additional CONSTRAINTS may apply, formulated using mechanisms such as:

types Cf. Carpenter 1992. As above, we shall assume Carpenter's well-typing, i.e., types restrict which features are appropriate on their instances, and features restrict their values to appropriate types.

relations Devices for describing not only feature structures (attribute-value matrices), but also relations between them. An example commonly cited to show the need for specifying relations is the three-place relation `append`:

$$\text{append}(a,b,c) \quad \leftrightarrow \quad a = \Lambda \text{ and } b = c \quad \text{OR} \tag{2}$$

$$a|\text{FIRST} = c|\text{FIRST} \text{ and } \text{append}(a|\text{REST}, b, c|\text{REST}) \tag{3}$$

where 'Λ' denotes the empty element, and '$\langle$value$\rangle$|path' is used to denote the value of $\langle$path$\rangle$ wrt $\langle$value$\rangle$. Lists are needed to describe linguistic phenomena which yield naturally to analyses as ordered sets, e.g., the daughters of a phrasal node or the subcategorization specifications of a head.

recursive type specifications These are type specifications on a type t which require an element of t' as value to a given attribute, where $t \sqsubseteq t'$. For example, an unsaturated sign is defined in HPSG as one with a nonempty SUBCAT—where SUBCAT is restricted to being a list of signs.

$$\left[\begin{array}{l} \textit{unsaturated-sign} \\ \text{SYN|LOC|SUBCAT} \langle [\ \textit{sign}\ ], \ldots \rangle \end{array}\right]$$

Similarly, the type *list* may be defined recursively (we assume [FIRST, REST] form) as either the empty list or a structure in which the feature REST is restricted to be of type *list*. A further example is the definition of *tree* as a *lexical-sign* or a *sign* whose attribute DAUGHTERS is a list of *tree*'s. We employ recursive type specifications for the representation of scopally underdetermined relationships.

Of course, recursion is not limited to the simple case where an immediate attribute of t requires a value of $t' \sqsupseteq t$. It suffices for any path $\text{ATTR}_1|\ldots|\text{ATTR}_n$ to impose such a restriction.

sets The motivation and technical development 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.) We employ the Pollard & Moshier notation of $\{*\ldots*\}$ as set delimiters.

6 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 is found in Moens et al. 1989 and Nerbonne 1991. 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 constraints which underspecify scope. 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.

6.1 Metalogical Task—Definition of Free Variables

We add to each type a feature FREE-VARS, which is constrained by type definitions. The feature 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. We suppress the full definition (found in Nerbonne 1992) in the interest of saving space, and provide one sample clause:

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

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

$$\textit{closed-wff} = \begin{bmatrix} \textit{wff} \\ \text{FREE-VARS } \{* \quad *\} \end{bmatrix}$$

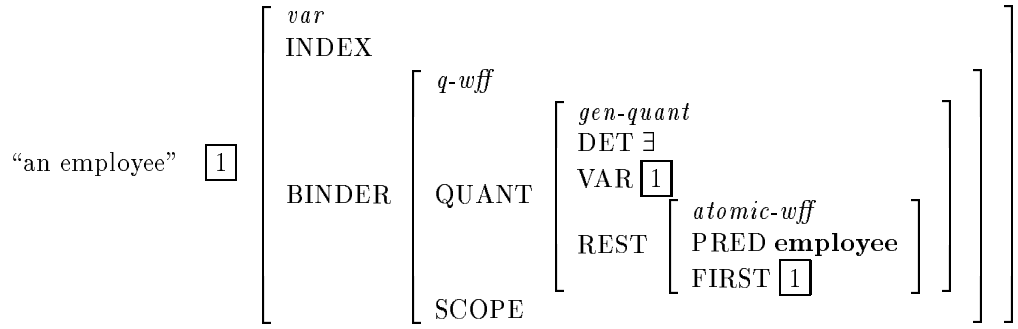
6.2 Overspecifying Logical Forms

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—in the case it describes well-formed formulas, the feature is redundant. 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:

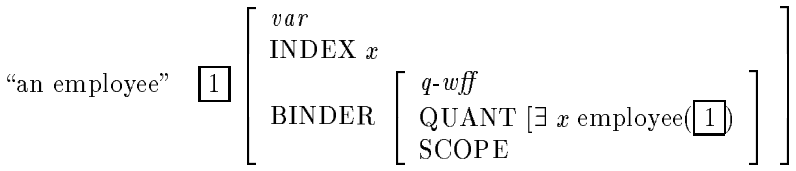
$$\langle \text{var} \rangle \ ::= \ x_0 \,|\, x_1 \,|\, \ldots \qquad \begin{bmatrix} \textit{var} \\ \text{INDEX} \\ \text{BINDER } [\textit{q-wff}] \end{bmatrix}$$

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 without 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.

6.3 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. 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.

An employee represents each department

(4)
$$\left[\begin{array}{l} \textit{atomic-wff} \\ \text{PRED } \textbf{represent} \\ \text{FIRST } \boxed{1} \left[\begin{array}{l} \textit{var} \\ \text{INDEX } x \\ \text{BINDER } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } [\exists x \ \textbf{employee}(\boxed{1}) \end{array}\right] \end{array}\right] \\ \text{SECOND } \boxed{2} \left[\begin{array}{l} \textit{var} \\ \text{INDEX } y \\ \text{BINDER } \left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT } [\forall y \ \textbf{department}(\boxed{2}) \end{array}\right] \end{array}\right] \end{array}\right]$$

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 (4) 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 (4) 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.

(4) 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.

6.4 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:

$$\left[\begin{array}{l} \textit{q-wff} \\ \text{QUANT} \\ \text{SCOPE } \boxed{1} \left[\text{ SCOPE* } \boxed{2} \right] \\ \text{SCOPE* } \{\boxed{1}, \boxed{2}\} \end{array}\right]$$

where '{' and '}' delimit feature description disjunctions.

We require that the feature SCOPE* is defined wherever SCOPE is (on *q-wff*), and that it always satisfy the disjunction here. 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 SCOPE* 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:

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

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 disambiguities 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*, 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. We choose the existential quantifier first here. Then 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.[4] It is clear that the characterization above crucially involved Pollard and 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 further notions: first, a restriction to closed formulas, as defined above. Second, we need a way to ban vacuous quantification, as provided in Nerbonne 1992.

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.

A second form of scope disambiguation involves "nested quantifiers", which are treated in Nerbonne 1992. Several further 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 1991); third, it would be most interesting to identify, represent and exploit disambiguating factors; fourth, it would be worth one's while to explore the semantic use of underspecified representations which are not actually subjected to disambiguation (cf. Poesio 1991 for a proposal for a semantic representation scheme in which scopes are unspecified); and fifth, the incorporation of syntactic restrictions on scope (e.g., scope islands) should be investigated.

7 Conclusions

The program of encoding semantics in the same formalism as syntax, found in a number of current computational linguistics frameworks, must face the issue of the expressive strength of the formalisms required. On the one hand, computational formalisms are preferably kept weak enough for descriptions (or programs) written in them to guaranteed to be tractable; failing tractability, to be decidable; and failing even decidability, at the very least to be complete. Semantics formalisms, on the other hand, need to deal with the inherent higher-order facilities of natural language meanings—e.g., higher order quantification ('MOST'),

[4] For the purposes of feature theory, it is worth noting that we genuinely need the set construct in order to describe the quantifiers involved in a logical form. Had we attempted to provide the specifications using, e.g., disjunction, we would have had insufficiently exact control over quantifiers in subformulas. In that case the disjunction could be satisfied, e.g., by setting the top-level QUANT value equal to one of the quantifiers involved—leaving the others free to vary arbitrarily.

type-raising, and even intensionality. These make complete logics impossible, and render chimerical goals such as decidability and tractability. Under these circumstances a strategy is appropriate which distinguishes formalisms by their computational properties, and provides for interfaces between them (rather than attempting to reduce all specification to the least tractable formalism).

In this context we propose to distinguish grammatical and semantic formalisms, to provide for an interface by using the former as a metalinguistic specification of the latter, and integrate (some) processing by compiling semantic descriptions even during grammatical processing. The strategy is appropriate not only for the syntax/semantics interface, but shows promise in dealing with nonsyntactic constraints on semantics, such as those arising in phonology and pragmatics. It furthermore allows the statement of generalizations which bridge syntax and semantics, such as those for which HPSG is admired.

An elaboration of this concept was introduced as illustration, and a sketch of semantic underspecification (relative quantifier scope) was provided, which moreover eliminates the need for auxiliary levels of logical form. Nerbonne 1991 provides further elaboration on the issue of disambiguation. On the negative side 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.

References

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

Barwise, J. 1989. Conditionals and Conditional Information. In *The Situation in Logic*, ed. J. Barwise, 97–136. Stanford: CSLI.

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

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

Blackburn, P. 1991. Modal Logic and Attribute Value Structures. In *Constraint Propagation, Linguistic Description, and Computation*, ed. M. Rosner, C.J.Rupp, and R. Johnson, 1–19. Lugano: Instituto Dalle Molle IDSIA.

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

Carpenter, B., C. J. Pollard, and A. Franz. 1991. The Specification and Implementation of Constraint-Based Unification Grammar. In *Proceedings of the Second International Workshop on Parsing Technology*. Cancun.

Creary, L. G., and C. J. Pollard. 1985. A Computational Semantics for Natural Language. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, 172–179.

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

Dowty, D. 1991. Thematic Proto-Roles and Argument Selection. *Language* 67(3).

Ebbinghaus, H.-D., J. Flum, and W. Thomas. 1978. *Einführung in die Mathematische Logik*. Darmstadt: Wissenschaftliche Buchgesellschaft.

Fenstad, J. E., P.-K. Halvorsen, T. Langholm, and J. van Benthem. 1987. *Situations, Language, and Logic*. Dordrecht: Reidel.

Gawron, J. M., and S. Peters. 1990. *Anaphora and Quantification in Situation Semantics*. Stanford University: CSLI Lecture Notes.

Gazdar, G., G. K. Pullum, R. Carpenter, E. Klein, T. Hukari, and R. D. Levine. 1987. Category Structures. Technical Report CSLI-87-102, CSLI.

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

Jackendoff, R. 1972. *Semantics Interpretation in Generative Grammar.* Cambridge: MIT Press.

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

Kaplan, D. 1979. On the Logic of Demonstratives. In *Contemporary Perspectives in the Philosophy of Language*, ed. P. A. French, J. Theodore E. Uehling, and H. K. Wettstein, 401–412. Minneapolis: University of Minnesota Press.

Kaplan, R., and J. Maxwell. 1988. An Algorithm for Functional Uncertainty. In *Proceedings of Coling 1988*, 303–305. Budapest.

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

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

Krifka, M. 1991. A Compositional Semantics for Multiple Focus Constructions. draft.

Moens, M., J. Calder, E. lein, M. Reape, and H. Zeevat. 1989. Expressing Generalizations in Unification-Based Grammar Formalisms. In *Proceedings of the 4th Meeting of the European Chapter of the Association for Computational Linguistics*, 174–181.

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

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

Nerbonne, J. 1991. Feature-Based Disambiguation. In *Constraint Propagation, Linguistic Description and Computation*, ed. R. Johnson, M. Rosner, and C.J.Rupp, xxx–yyy.

Nerbonne, J. 1992. A Feature-Based Syntax/Semantics Interface. In *Proceedings of the Second International Conference on the Mathematics of Language*, ed. A. Manaster-Ramer and W. Zadrozny, xxx–yyy. Annals of Mathematics and Artificial Intelligence.

Poesio, M. 1991. Relational Semantics and Scope Disambiguation. In *Proceedings of the Second Conference on Situation Theory and Its Applications*, ed. J. Barwise, J. M. Gawron, and G. Plotkin. Stanford University: CSLI Lecture Notes.

Pollard, C. 1989. The Syntax-Semantics Interface in a Unification-Based Phrase Structure Grammar. In *Views of the Syntax-Semantics Interface: Proceedings of the Workshop "GPSG and Semantics", Technische Universität Berlin, 22-24.Feb 1989*, ed. S. Busemann, C. Hauenschild, and C. Umbach, 167–184. Technische Universität Berlin: KIT FAST.

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

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

Reape, M. 1991. An Introduction to the Semantics of Unification-Based Grammar Formalisms. Technical Report R3.2.A, DYANA, University of Edinburgh.

Roberts, D. 1991. Linking Thematic Roles and Syntactic Arguments in HPSG. Master's thesis, Cornell University.

Rooth, M. 1985. *Association with Focus.* PhD thesis, University of Massachusetts at Amherst.

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

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

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

Shieber, S., H. Uszkoreit, F. Pereira, J. Robinson, and M. 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, G. 1988. A Feature Logic with Subsorts. Technical Report 33, WT LILOG–IBM Germany.

von Stechow, A. 1989. Focusing and Backgrounding Operators. Arbeitspapier 6, Fachgruppe Sprachwissenschaft, Universität Konstanz.

Wechsler, S. M. 1991. *Argument Structure and Linking*. PhD thesis, Stanford University.

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