Lexicons for Feature-Based Systems

Gosse Bouma and John Nerbonne

Alfa-informatica and Cognitive and Behavioral Neurosciences
Rijksuniversiteit Groningen, The Netherlands
{nerbonne,gosse}@let.rug.nl

Abstract. Feature description languages including relational constraints and default inheritance allow lexical specifications which are transparent, concise, and capture all relevant generalizations; e.g., they enable the encoding of inflectional and derivational relationships. Inheritance-based approaches to inflectional morphology–made popular by formalisms such as DATR–can easily be integrated. This has the advantage that interface problems disappear.

Zusammenfassung. Merkmalsbeschreibungssprachen, die auch Default-Vererbung zulassen, ermoeglichen, daß sich lexikalische Spezifizierungen kompakt, transparent, und ausdruckskraeftig schreiben lassen. Sie erlauben z.B. die Beschreibung von Flexion und Derivation. Dieses deutet an, daß das Gebrauch von speziellen Lexikonbescriebungssprachen, wie DATR, schlecht motiviert ist. Weitere Schwierigkeiten tauchen auf, wo spezielle Systeme an Merkmalssystemen gekoppelt werden sollten.

1 Introduction

Feature-based grammars are employed nearly universally for the description of syntax and semantics in computational linguistics. Lexicons for feature-based grammars must provide a way of matching orthographic or phonetic form with the properties specified in a grammar in the form of feature descriptions. This paper argues that feature description languages—enhanced with relational constraints and default inheritance—are the proper vehicles for the expression of such relations. In particular, we argue that there are no reasons to adopt an approach in which feature-based systems are combined with a special purpose formalism–such as DATR–to support lexical description. The integrated approach avoids the interface problems that exist for systems that seek to combine two radically different formalisms. Also, inheritance-based decriptions of inflectional morphology–typical for DATR theories–can easily be integrated in feature-based formalisms, thus leading to analyses in which the strengths of the feature-based and inheritance-based approach can be combined in an optimal fashion.

2 Formal Preliminaries

Feature-Based grammar formalisms have made use of a wide range of formal operations and devices in order to obtain declarative, succinct, and elegant analyses of linguistic phenomena. For the development of feature-based lexicons, the introduction of nonmonotonic operations for combining feature structures and the introduction of relational constraints appears to be especially relevant. Other formal devices that have been used are types (Bird and Klein, 1993) and (distributed) disjunctive feature specifications (Krieger and Nerbonne, 1993). We introduce the use of nonmonotonic operations and relational constraints below.

Default Inheritance. The use of defaults or OVERWRITING is crucial for a practical lexical tool. The key advantage of default specifications is that they allow the description of SUBREGULARITIES, classes of items whose properties are largely, but not perfectly regular. In a system with default inheritance, these are not anomalous, but rather imperfectly regular. Such specifications are appropriate for the description of the innumerable linguistic phenomena which are partially regular—or, which, at the present stage of our linguistic knowledge, cannot be succinctly described in a completely regular way. Some examples of exceptional behavior for which default specification is appropriate include: irregular and missing inflectional forms, irregular derivational form or meaning (Krieger and Nerbonne, 1993), irregular syntax (Flickinger, 1987, pp.64-66); irregular subcategorization specifications (Flickinger and Nerbonne, 1992).

The use of lexical defaults is a fairly harmless form of nonmonotonicity, since the lexicon is nonmonotonic only with respect to lexical development—the use of such defaults leads to none of the problems associated with nonmonotonic reasoning; e.g., inferences about phrases never need to be retracted, and the NL system may be configured to be perfectly monotonic at run-time.

Monotonic (template) inheritance has been implemented in feature-based systems by means of unification. Nonmonotonic inheritance can be implemented using default unification only (Bouma, 1992) or using default unification combined with some special mechanism to account for multiple default inheritance (Russell et al., 1992).

Default unification (Bouma, 1992; Carpenter, 1992) (or priority union (Calder, 1991)) is an operation which combines two feature structures nonmonotonically, in such a way that information from the default argument is only added to the strict argument in as far as this information is compatible with that already specified in the strict argument. Thus, default unification is a non-symmetric operation, which (unlike unification) never fails, and which always produces a result which is at least as specific as the strict argument.

The precise definition of default unification has been the subject of some debate (see also Copestake (1992)). One of the issues is that for feature structures containing reentrancies, there are in general several possibilities to resolve potential unification clashes. Carpenter (1992) and Calder (1991) choose to give (the generalization of) the disjunction of all solutions as the result of default unification, whereas Bouma (1992) removes default reentrancies if they refer to nodes already defined in the strict argument. We prefer the latter definition, as it does not produce disjunctive or overly general results.

Relational Constraints. Grammar formalisms such as HPSG have made extensive use of list specifications in order to encode, e.g., phonological realization (PHON) or subcategorization requirements (SUBCAT). Naturally, once one uses lists, common operations for lists become relevant, such as deletion/insertion, concatenation or reversing. For instance, the rule schemata of HPSG require that the value of PHON of a mother node be the concatenation of the PHON values of the daughters, and that the SUBCAT value of the head daughter is the concatenation of the list of non-head daughters and the value of the mother's SUBCAT.

List operations have also been used in the specification of lexical elements. In (Hinrichs and Nakazawa, 1993), who present an analysis of verb clustering in German, the SUBCAT value of modal verbs is specified as being the concatenation of a list containing a subject NP and a verbal complement with the value of SUBCAT specified on this verbal complement. To express constraints on feature structures of this kind, relational constraints need to be introduced. For instance, the Hinrichs-Nakazawa analysis of modals could be encoded as follows:

(1)
$$\begin{bmatrix} \text{HEAD} & \begin{bmatrix}\text{MAJ} & \text{V}\end{bmatrix} \\ \text{SUBCAT} & \left\langle \text{NP-NOM}, \begin{bmatrix}\text{SUBCAT}\ \boxed{1}\end{bmatrix} \right\rangle \ \&\ \boxed{1} \end{bmatrix}$$

where '&' denotes the append operation on lists. Making the relational constraint explicit, we could also write this as:

(2)
$$\begin{bmatrix} \text{HEAD} & \begin{bmatrix}\text{MAJ} & \text{V}\end{bmatrix} \\ \text{SUBCAT} & \boxed{1} \end{bmatrix} \leftarrow \mathit{append}\left(\left\langle \text{NP-NOM}, \begin{bmatrix}\text{SUBCAT} & \boxed{2}\end{bmatrix} \right\rangle, \boxed{2}, \boxed{1} \right)$$

Relational constraints are of importance in the area of phonology and morphology. They can be used to specify the relation between the list of morphemes that constitute a word and the *surface* realization of this word (i.e. its phonological or orthographic form), thus providing a method to incorporate relational (two-level) approaches to phonology and orthography (Trost and Matiasek, 1994). Furthermore, as we will show below, they can be used to specify the connection between the morphological form of a word and its morphosyntactic feature-specification, thus eliminating the need for morphological rules.

Finally, relational constraints open the way to recursive formulations of the sort needed for defining some derivational relationships. Thus Matthews (1974) notes examples such as *institut(e)+<u>tion</u>+al+iz(e)+a<u>tion</u>*, where the deverbal nominalizing *-tion* has applied twice. A simpler example is *anti-anti-missile.* Miller (1992) proposes recursive rule which places adjectives in the nouns subcategorization list. Furthermore, it is commonly observed that hierarchical structure is found in word formation, proceeding from the structural ambiguity of examples such as *[un- [do -able]]* as opposed to *[[un- do] -able]*. A natural formulation accounting for these properties is a simultaneous recursion, e.g., a set of rules such as $V \rightarrow Adj$, $Adj \rightarrow N$, $N \rightarrow V$.

Both recursion and hierarchical structure are difficult to model when only inheritance is employed. In that case there cannot be any difference between inheriting properties one or more times (idempotency), which would suggest that *anti-anti-missile* ought to be the same as *anti-missile*. Likewise, given the associativity of inheritance, we should have no means of explaining structural ambiguity on the basis of an inheritance model. Finally, simple examination of derivationally complex words (e.g., *un* + *necessary*) suggests the COMPONENTS do not simply share properties with complex words, as an "inheritance" treatment might suggest. All of these points suggest that however successful the inheritance analysis of some derivational phenomena is (Russell et al., 1992; Kilgariff, 1993), we shall never complete a theory of derivation on this basis.

3 Building Feature-Based Lexicons with DATR

A number of special-purpose lexical inheritance mechanisms have been proposed to support the construction of lexicons for unification-based grammar formalisms. DATR (Evans and Gazdar, 1989; Evans and Gazdar, 1990) has been particularly important for providing a rigorous declarative definition of default inheritance and because it has been used in several applications in combination with feature-based grammars (Cahill and Evans, 1990; Kilbury, Naerger, and Renz, 1991; Andry et al., 1992).[1]

A DATR theory consists of a collection of classes, defining a set of path equations and inheritance links. An example is provided in (3) below.

(3) **VERB:** ⟨syn aux⟩ == no
⟨syn cat⟩ == V
⟨morph pres fin⟩ == ("⟨morph root⟩")
⟨morph pres fin sing 3⟩ == ("⟨morph root⟩" *s*)
⟨morph past⟩ == ("⟨morph root⟩" *ed*)
⟨morph pres part⟩ == ("⟨morph root⟩" *ing*).

COME : ⟨⟩ == **VERB**
⟨morph root⟩ == *come*
⟨morph past⟩ == *came*
⟨morph past part⟩ == ⟨morph root⟩.

The denotation of a class is a tree, with labelled edges and nodes. **COME**, for instance, denotes the tree in Fig. 1. The tree can be used to determine the morphological forms of *come*. To find the form of the plural present tense, for instance, go to the corresponding node in the tree (which can be found by following the path ⟨morph pres fin plur⟩), and then see if this node has a label. If so, this is the requested form, if not, one must go up in the tree to the nearest node that has a label (⟨morph pres fin⟩ in this case) to find the requested form. Note that because of this mechanism for locating values, DATR graphs must be trees, i.e. a given node must have a unique parent.

[1] We restrict our attention here to that part of DATR treated in the metatheory (Evans and Gazdar, 1989), but excluding evaluable paths (which Moser (1992) has shown to lead to Turing-equivalence).

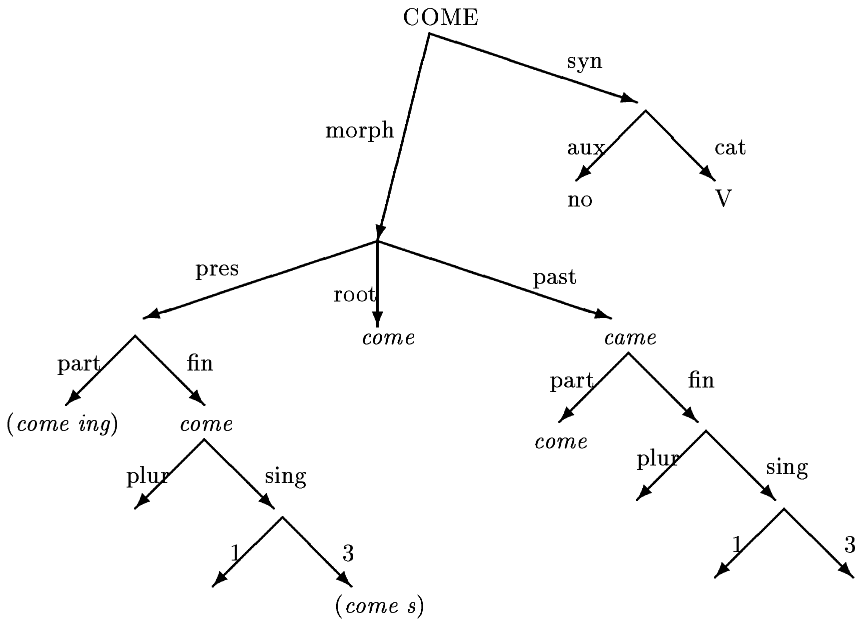

Fig. 1. The DATR tree for the lexeme *come*.

Interfacing DATR with Feature-Based Theories. Although both feature description formalisms and DATR have an interpretation as graph description languages, their interface is not straightforward.

First, structure-sharing via reentrancies has turned out to be a crucial aspect of feature-based accounts of agreement, semantics, control, etc. In DATR there is no way to denote structure-sharing directly. It is possible to use a class node (such as **VERB**) in the definition of several subclasses, but this does only imply that all the information present in superclass is in fact shared by the subclass, as locally supplied values may block the inheritance of inherited values.

The construct that is superficially most similar to the path equations used for expressing reentrancy in a feature-based formalism is the path equivalence statement. The semantics of such statements does not correspond to reentrancy (i.e. structure-sharing), however, but to assigning (asymmetrically) a value to a node. For instance, the value of ⟨A C⟩ in (4) is FOO, but the value of ⟨B C⟩ is BAR. Thus, ⟨B⟩ == ⟨A⟩ does not express that the subgraphs rooted at A and B are identical, as is the case for a reentrancy statement in a feature-based system.

(4)
N: ⟨A⟩ == FOO
⟨B⟩ == ⟨A⟩
⟨BC⟩ == BAR.

Secondly, even though both DATR and feature formalisms are interpretable as graph-description languages, their MODELING CONVENTIONS are radically different. DATR uses relative depth to model default preference, while feature formalisms attach no significance to the relative depth (number of edges) between nodes. Since the default preferences concern the forms of words, we group attributes in DATR where they tend toward syncretism (sharing a form). In feature formalisms the modeling conventions are rather different, leading one to group attributes (e.g., PER and NUM under AGR) where a grammatical process (agreement) makes common reference to them. In general, grammatical processes do not follow the lines of morphological form.

Kilbury, Naerger, and Renz (1991) show how to construct an interface from DATR to feature formalisms (using the example of PATR-II). They advocate constructing an interface by using DATR as a metalanguage in which one can describe and constrain (the syntax of) PATR-II equations. It is unsatisfying for two reasons: first, it ignores underlying semantics at the cost of having to write specifications about PATR-II syntax, including bracketing, use of colon, etc., an untidy level of indirection. Second, in using DATR as a metalanguage, massive redundancy is introduced into lexical specifications, which now specify not only form values for the attribute '⟨PERSON⟩', but attributes in which the PATR-II value 'PERSON' figures (as DATR "PERSON"). Formally, these have nothing to do with one another. The name of the DATR attribute and (part of) the value of a completely distinct attribute look similar to human eyes, but are formally unrelated. These problems reflect the general mismatch between DATR and feature-based formalisms in expressive capacity.

Finally, and in spite of Kilgariff's (1993) interesting work modelling some derivational relations in the pure inheritance machinery of DATR, we know of no work attempting to model potentially recursive derivational relations, and we remain sceptical about relying on inheritance alone for this (cf. § 2 above).

4 Incorporating Inheritance-Based Morphology

The inheritance-based approach to morphology promoted by DATR does not require that one adopts DATR's default inheritance mechanism. We can incorporate an inheritance-based analysis of inflection, using feature structures as the only data-type, and default unification to model nonmonotonic inheritance. Consider, for instance, how the DATR class **VERB** (as defined in (3)) could be encoded in a feature-based formalism:

(5)

$$\textbf{VERB} \equiv \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{AUX} & \text{NO} \\ \text{MAJ} & \text{V} \end{bmatrix} \\ \text{MORPH} & \begin{bmatrix} \text{ROOT} & \boxed{1} \\ \text{PRES} & \begin{bmatrix} \text{FIN} & \begin{bmatrix} \text{-SG3} & \boxed{1} \\ \text{SG3} & \left\langle \boxed{1},\text{S} \right\rangle \end{bmatrix} \\ \text{PART} & \left\langle \boxed{1},\text{ING} \right\rangle \end{bmatrix} \\ \text{PAST} & \begin{bmatrix} \text{FIN} & \boxed{2} \left\langle \boxed{1},\text{ED} \right\rangle \\ \text{PART} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

For the purpose of illustration, we follow the DATR encoding almost literally, except for the fact that implicit node labelling must be avoided, as a feature structure cannot have an atomic value and at the same time have outgoing edges. To avoid redundancy, we therefore use a feature -SG3 to cover all agreement values other than third person singular.

Next, consider the translation of the class **COME**, which inherits nonmonotonically from **VERB** in the DATR example. In the feature-based approach we can define **COME** using default unification ($\sqcup$ denotes unification and $\sqcup_!$ default unification):

(6)
$$\mathbf{COME} \equiv \left(\mathbf{VERB} \sqcup \begin{bmatrix}\textsc{root} & \textsc{come}\end{bmatrix}\right) \sqcup_! \begin{bmatrix}\textsc{morph} \mid \textsc{past} & \begin{bmatrix}\textsc{fin} & \textsc{came}\\ \textsc{part} & \textsc{come}\end{bmatrix}\end{bmatrix}$$

which is equivalent to:

(7)
$$\mathbf{COME} \equiv \begin{bmatrix}\textsc{syn} & \begin{bmatrix}\textsc{aux} & \textsc{no}\\ \textsc{maj} & \textsc{v}\end{bmatrix}\\ \textsc{morph} & \begin{bmatrix}\textsc{root} & \boxed{1}\ \textsc{come}\\ \textsc{pres} & \begin{bmatrix}\textsc{fin} & \begin{bmatrix}\textsc{-sg3} & \boxed{1}\\ \textsc{sg3} & \left\langle \boxed{1}, \textsc{s}\right\rangle\end{bmatrix}\\ \textsc{part} & \left\langle \boxed{1}, \textsc{ing}\right\rangle\end{bmatrix}\\ \textsc{past} & \begin{bmatrix}\textsc{fin} & \textsc{came}\\ \textsc{part} & \textsc{come}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

The value of the paths ⟨MORPH PAST FIN⟩ and ⟨MORPH PAST PART⟩ is overwritten nonmonotonically with atomic values. There is a slight difference in the feature-based encoding in that it does not make explicit the reentrancy between the root and the past participle form suggested by the DATR equation ⟨MORPH PAST PART⟩ = ⟨MORPH ROOT⟩. We do not make the reentrancy explicit as part of the strict information, as this would imply that all the default reentrancies between ⟨MORPH ROOT⟩ and other parts of the feature structure would be lost.[2]

As in the DATR theory, the feature structures just given specify only the various morphological forms of a lexeme, not how inflected forms are related to features for agreement, tense, etc. However, since the feature structures do specify grammatical objects, the inflected forms can be derived by means of lexical rules such as (8). As paradigmatic information is part of the feature structure of lexemes, only one lexical rule is needed for each slot in the verbal paradigm.

[2] This treatment of the past participle form is satisfying if it holds for only this verb. If one wants to capture a generalization about all verbs whose past participle form does not end in -ED, but instead is identical to that of their root, one can specify the strict value of ⟨MORPH PAST PART⟩ to be $\langle \top, \emptyset \rangle$. This value overwrites the tail of the list supplied as value for ⟨MORPH PAST PART⟩ (as the empty suffix) but leaves the reentrancy between the head of list and root intact.

(8)

$$\mathbf{SG3\text{-}LR} \equiv \begin{bmatrix} \textsc{syn} & \boxed{1} \begin{bmatrix} \textsc{maj} & \textsc{v} \\ \textsc{agr} & \begin{bmatrix} \textsc{per} & 3 \\ \textsc{num} & \textsc{sg} \end{bmatrix} \end{bmatrix} \\ \textsc{phon} & \boxed{2} \end{bmatrix} \Rightarrow \begin{bmatrix} \textsc{syn} & \boxed{1} \\ \textsc{morph} & \begin{bmatrix} \textsc{pres} \mid \textsc{fin} \mid \textsc{sg} & \boxed{2} \end{bmatrix} \end{bmatrix}$$

A Relational Approach. Paradigmatic information can be included in feature structures in a fashion that resembles the method used in DATR fragments. However, within HPSG it has been proposed that (inflectional) morphology can be described using (distributed) disjunctions (Krieger and Nerbonne, 1993) or relational constraints (Kathol, 1994), thus avoiding the use of lexical rules altogether. Such an approach need not be in conflict with inheritance-based methods, as both can be fruitfully combined.

In the relational approach, lexical entries are defined using relational constraints that will make reference (either explicitly or implicitly) to the paradigm of the class to which the lexical entry belongs. A lexical entry can be thought of as an underspecified feature structure that can be instantiated by resolving the constraints for the given entry. Each solution will give a different inflected form.

As an example, consider once more the inflection of regular verbs in English. In a relational approach, the various forms of a regular verb are defined as follows:

(9)

$$\mathbf{KISS} \equiv \boxed{1} \begin{bmatrix} \textsc{root} & \textsc{kiss} \\ \textsc{syn} & \begin{bmatrix} \textsc{maj} & \textsc{v} \end{bmatrix} \end{bmatrix} \leftarrow \mathit{member}\big(\boxed{1}, \mathbf{VERB}\big)$$

The constraint *member* specifies that the feature structure of **kiss** must unify with one of the entries in the paradigm **VERB**. An entry in the paradigm (which is now a feature structure consisting of a list) specifies the phonological form of a verb as well as the corresponding morphosyntactic features:

(10)

$$\mathbf{VERB} \equiv \left\langle \begin{bmatrix} \textsc{phon} & \langle \boxed{1}, \textsc{s} \rangle \\ \textsc{mor} & \begin{bmatrix} \textsc{root} & \boxed{1} \end{bmatrix} \\ \textsc{syn} & \begin{bmatrix} \textsc{agr} & \textsc{sg}3 \end{bmatrix} \end{bmatrix}, \ldots, \begin{bmatrix} \textsc{phon} & \langle \boxed{1}, \textsc{ed} \rangle \\ \textsc{mor} & \begin{bmatrix} \textsc{root} & \boxed{1} \end{bmatrix} \\ \textsc{syn} & \begin{bmatrix} \textsc{tns} & \textsc{pst} \\ \textsc{form} & \textsc{fin} \end{bmatrix} \end{bmatrix}, \begin{bmatrix} \textsc{phon} & \langle \boxed{1}, \textsc{ed} \rangle \\ \textsc{mor} & \begin{bmatrix} \textsc{root} & \boxed{1} \end{bmatrix} \\ \textsc{syn} & \begin{bmatrix} \textsc{form} & \textsc{psp} \end{bmatrix} \end{bmatrix}, \ldots \right\rangle$$

Nonmonotonic inheritance can still be used to implement irregular paradigms. The various forms of the lexeme *come*, for instance, are not obtained using **VERB**, but using the result of default-unifying this paradigm with the strict information supplied in the definition of **come**. The strict information is a list of the same length of the verbal paradigm, whose elements are unspecified except for the elements corresponding to the finite past tense and past participle form. The phonology of these elements is specified explicitly, thus overwriting the default value for PHON.[3]

[3] A similar proposal was made by Krieger and Nerbonne (1993), but since they specified paradigms as a disjunctive feature structure, the interaction with default unification was unclear. Defining paradigms as ordered lists avoids this problem.

(11) $$\mathbf{COME} \equiv \boxed{1}\begin{bmatrix}\text{ROOT} & \text{COME}\\ \text{SYN} & \begin{bmatrix}\text{MAJ} & \text{V}\end{bmatrix}\end{bmatrix} \leftarrow$$

$$member\left(\boxed{1}, \mathbf{VERB} \sqcup_! \left\langle \top, \ldots, \begin{bmatrix}\text{PHON} & \text{CAME}\end{bmatrix}, \begin{bmatrix}\text{PHON} & \text{COME}\end{bmatrix}, \ldots \right\rangle\right)$$

One of the advantages of the relational approach is that a lexicon in which inflected forms can be found by resolving constraints is that it can be used not only to instantiate the morphological features of an infelcted form, but also to determine what the form of a lexeme must be, given a specification of its morphological features.

5 Conclusions

Lexical specification can be done directly in feature description formalisms, provided such formalisms include nomonotonic operations on feature structures, relational constraints, types, etc. With the introduction of such powerful devices in feature-based formalisms, the original motivation for introducing special-purpose formalisms, such as DATR, has lost its force. The examples we have provided in the previous section show that the inheritance-based approach to morphology can be incorporated in feature-based formalisms in such a way that rule-based as well as relational analyses of morphology can be supported. We conclude, therefore, that the advantages of using nonmonotonic inheritance in morphology in combination with feature-based grammars can only be exploited fully if the two are part of one integrated formalism.

References

Andry, François, Norman Fraser, Scott McGlashan, Simon Thornton, and Nick J. Youd. 1992. Making DATR work for speech: Lexicon compilation in SUNDIAL. *Computational Linguistics*, 18(3):245–268.

Bird, Steven and Ewan Klein. 1993. Enriching HPSG phonology. EUCCS RP-56, University of Edinburgh.

Bouma, Gosse. 1992. Feature structures and nonmonotonicity. *Computational Linguistics*, 18(3):185–205.

Cahill, Lynne and Roger Evans. 1990. An application of DATR: The TIC lexicon. Cognitive Science Research Reports, , number SCRP 139. School of Cognitive and Computing Sciences, University of Sussex, pages 31–39.

Calder, Jonathan. 1991. *An Interpretation of Paradigmatic Morphology.* Ph.D. thesis, University of Edinburgh.

Carpenter, Bob. 1992. Skeptical and creduluous default unification with applications to templates and inheritance. In Ted Briscoe, Anne Copestake, and Valerie de Paiva, editors, *Default Inheritance within Unification-Based Approaches to the Lexicon.* Cambridge University Press, Cambridge.

Copestake, Anne. 1992. Defaults in the LRL. In Ted Briscoe, Anne Copestake, and Valerie de Paiva, editors, *Default Inheritance within Unification-Based Approaches to the Lexicon*. Cambridge University Press, Cambridge.

Evans, Roger and Gerald Gazdar. 1989. Inference in DATR. In *Proc. of the 4th European ACL*, pages 66–71.

Evans, Roger and Gerald Gazdar. 1990. The DATR papers. Technical Report SCRP 139, School of Cognitive and Computing Sciences, University of Sussex.

Flickinger, Daniel. 1987. *Lexical Rules in the Hierarchical Lexicon*. Ph.D. thesis, Stanford University.

Flickinger, Daniel and John Nerbonne. 1992. Inheritance and complementation: A case study of *easy* adjectives and related nouns. *Computational Linguistics*, 19(3):269–309.

Hinrichs, Erhard and Tsuneko Nakazawa. 1993. Linearizing AUX in german verbal complexes. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German Grammar in HPSG*. CSLI, Stanford.

Kathol, Andreas. 1994. Agreement and the syntax/morphology interface in HPSG. In Robert Kasper and Robert Calcagno, editors, *OSU Working Papers in Linguistics, Vol. 45*. Linguistics, Ohio State University, Columbus, Ohio.

Kilbury, James, Petra Naerger, and Ingrid Renz. 1991. DATR as a lexical component for PATR. In *Proc. of the 6th Meeting of the European Chapter of the Association for Computational Linguistics*, pages 137–142.

Kilgariff, Adam. 1993. Inheriting verb alternations. In *Proc. of the 7th Meeting of the European Chapter of the Association for Computational Linguistics*, pages 213–221.

Krieger, Hans-Ulrich and John Nerbonne. 1993. Feature-based inheritance networks for computational lexicons. In Ted Briscoe, Valerie de Paiva, and Anne Copestake, editors, *Inheritance, Defaults, and the Lexicon*. Cambridge University Press, Cambridge, pages 90–136. Also DFKI Research Report RR-91-31.

Matthews, P.H. 1974. *Morphology*. Cambridge, England: Cambridge University Press.

Miller, Philip. 1992. *Clitics and Constituents in Phrase Structure Grammar*. Outstanding Dissertations in Linguistics. New York: Garland. (Utrecht Diss., 1991).

Moser, Lionel. 1992. Simulating turing machines in DATR. Cognitive Science Research Reports SCRP 241, School of Cognitive and Computing Sciences, University of Sussex.

Russell, Graham, Afzal Ballim, John Carroll, and Susan Warwick-Armstrong. 1992. A practical approach to multiple default inheritance for unification-based lexicons. *Computational Linguistics*, 19(3):311–337.

Trost, Harald and Johannes Matiasek. 1994. Morphology with a null-interface. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, Kyoto.