New Lexical Entries for Unknown Words

James Kilbury, Petra Naerger, Ingrid Renz[*]

Abstract

The following paper presents an approach for simulating the acquisition of new lexical entries for unknown words, an issue that is central to natural language processing since no lexicon can ever be complete. Acquisition involves two main tasks. First, the appropriate information about an unknown word in a given linguistic context (i.e. sentence) is identified. It is shown that this task requires new general considerations about shared information in unification-based representations. Second, the collected information is formulated in a new lexical entry according to a comprehensive theory of the lexicon which defines the form of lexical entries and the relations between them. This task is solved by a general algorithm that depends only on the form of the collected information and is independent of the content, i.e. treats all unknown words in the same way.

1. General issues concerning unknown words

The lexicon of any natural language must be viewed as an open system in which items are constantly added, modified, and deleted. Various investigators in natural-language processing (NLP) (cf. the contributions in Zernik 89) have noticed this property but treated it basically as a shortcoming of language that needs to be overcome. In contrast, we view this property as an inherent and essential aspect of natural language. Consequently, the lexicon of an NLP

[*] Heinrich-Heine-Universität Düsseldorf, Germany. The presented work was carried out in the project "SIMLEX - Simulation lexikalischen Erwerbs" (simulation of lexical acquisition) funded by the Deutsche Forschungsgemeinschaft. For helpful comments and discussions on the topic we would like to thank Gosse Bouma, Lynne Cahill, Roger Evans, Gerald Gazdar, and Dafydd Gibbon.

system[1] should reflect this characteristic so that the model itself encompasses incompleteness. Thus, when a sentence is parsed the problem may arise that no adequate entries for certain words can be found in the given lexicon, but the system must nevertheless be able to parse such a sentence. Better yet, it should utilize the sentence as a source of new information in order to extend the lexicon.

There are various reasons why an adequate entry for a given word in an analyzed sentence may not be found. The most obvious one is that no entry exists for the word. In this case the system must check whether the word is phonetically and morphologically admissible in the given context. If it is admissible and not to be interpreted as an error (e.g. a misspelling), then the form is a "new" or "unknown" word. Even if a lexical entry exists, it may not be appropriate for this context since one word form may be associated with different properties, not all of which are recorded in the lexicon. The syntactic category is such a property. The German word form *weichen*, for example, is a verb ('withdraw') as well as a noun ('shunts') and an adjective ('soft'). Here the words are not related semantically, whereas for the German word form *essen* (verb 'eat' or noun 'meal') a semantic relation exists. Even if the category of the lexical entry is correct for the context, other properties may be inappropriate. For example, one verb may have different subcategorization frames, some of which are missing in the lexical entry. In our approach all admissible words for which no suitable lexical entry is found are viewed as unknown words.

Many parsers simply fail upon encountering new words, and a standard parsing algorithm must be adapted to deal with them in the first place. In particular, the context of a new word must be used to assign it information which, if possible, allows the sentence containing it to be parsed; in some cases no such assignment of information to the new word allows a successful analysis. After successful parsing there are two possibilities for the further treatment of such unknown words. Either no further actions are carried out and the lexicon remains unchanged, or information about the unknown words is collected and used to formulate new lexical entries. In this case new lexemes are acquired and the lexicon is correspondingly changed. It should be clear that the first possibility is unsatisfactory since the model then fails to capture one of the main properties of lexicons.

[1] At this point 'lexicon' and 'lexical entry' are pretheoretical concepts and do not presuppose any particular linguistic theory. In section 2 a theoretical characterization of the lexicon will be developed.

The acquisition of new lexemes makes it necessary to update the lexicon in such a way that the new lexemes are integrated in the existing lexical structure. The process of integration differs depending on the respect in which the word is unknown. Words for which an entry, though inappropriate, exists require a different treatment than those for which no entry is found. In the former case the existing entry must be revised according to the new information. In the latter case a new lexical entry must be created; this constitutes the main issue of the paper. Here, the structure of lexical entries as well as the overall structure of the lexicon into which the new entry has to be integrated are crucial.

2. Structure of the lexicon and the lexical entries

In an NLP system lexical entries must have a consistent and well-defined form. Within unification-based frameworks (cf. Shieber 86) they can be defined with templates or directly with path equations. The established structure holds not only for existing entries but also for new lexical entries since the latter have to be integrated into the current lexicon. Furthermore, this structure and the overall structure of the lexicon influence the way in which new lexical entries are formulated. So first of all the structure of the lexicon and the lexical entries have to be specified.

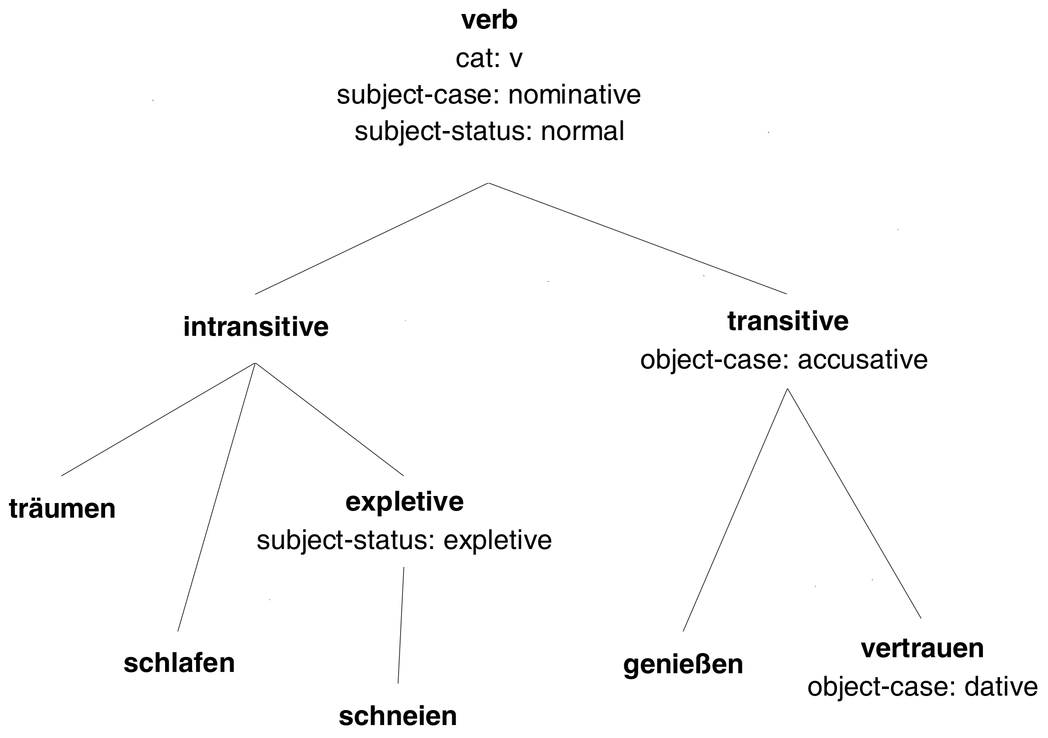
An adequate representation for a lexicon must meet certain conditions. First of all, general requirements like computational tractability, user friendliness, and reduction of redundancy have to be met. Whereas, for example, lexical entries directly defined with path equations are computationally tractable, they are highly redundant and not easily readable for human users and are therefore unsuitable. Templates on the other hand fulfill these general requirements, but fail to model the following aspects typical of the lexicon.

Characteristically, lexical entries are not isolated items but rather are related to other linguistic objects in a hierarchy (cf. the pioneering work in Flickinger/Pollard/Wasow 85). In this hierarchy various relationships hold between the items, including not only regularities but also subregularities and exceptions that cannot be captured by templates[2] . Figure (1) illus-

[2] The representation of these relationships requires nonmonotonic devices which are not included in standard unification formalisms. A well-defined nonmonotonic extension is presented in Bouma 90.

trates these kinds of relationships in a lexicon.

(1) hierarchical relationships in a lexicon



In the above example the following dependencies are expressed: verbs belong to the category v and subcategorize for a normal nominative subject; intransitive verbs are regular verbs, i.e. exhibit all properties of verbs; on the other hand, expletive verbs are irregular intransitive verbs since they take the expletive subject *es* but are intransitive verbs in all other respects; transitive verbs are regular verbs which subcategorize for an object in accusative case as well as a subject. As for the lexemes themselves, *träumen* ('dream') and *schlafen* ('sleep') are regular intransitive verbs, *schneien* ('snow') is an expletive verb, *genießen* ('enjoy') is a regular transitive verb, and *vertrauen* ('trust') is basically a transitive verb but takes a dative object.

These different kinds of dependencies have to be characterized using the expressive devices of an appropriate representation language. One such language is DATR, which was developed especially for this purpose by Roger Evans and Gerald Gazdar (see Evans/Gazdar 89a, 89b, 90) and represents a restricted class of semantic networks. It includes seven inference rules and a default inheritance mechanism which operate on well-formed DATR theories.

A DATR theory (net) is defined by a set of statements. Every statement consists of a node-path pair with an associated value. This value is either stated directly or inherited from another source (another node, path, or node-path pair). Some well-formed statements are listed in (2).[3]

(2) DATR statements

```
NODE1:    <path1> == av.
NODE1:    <path2> == <path1>.
NODE2:    <path1> == NODE1.
NODE2:    <path2> == NODE1:<path1>.
```

The information represented in a DATR net can be accessed by queries, whose evaluation is determined by the inference rules and the default mechanism mentioned above. Such a query consists of a node-path pair, and its evaluation returns a value which is either atomic or a sequence of values. All sensible queries concerning the above net return the atomic value **av**.

Despite their syntactic similarities, DATR and unification-based grammar formalisms like PATR-II are quite different in their semantics, so that a bridge between the two must be available in order to use a lexicon encoded in DATR. Various possibilities for providing such a bridge are described in Kilbury/Naerger/Renz 91. The strategy adopted is to make the returned values be the usual feature structures required by the PATR system. In this approach the task of DATR is twofold: First, it represents the lexical information according to linguistic generalizations. Second, DATR means are exploited to produce the feature structures required by the PATR system. In this way the lexicon is integrated into the whole system and the organization of the feature structures is imposed by the DATR theory.[4]

[3] The examples are not exhaustive since several DATR devices are not mentioned here. Note also that there is a convention for abbreviations: for a set of DATR statements belonging to the same node the node name has to be stated only once, as in the following example:

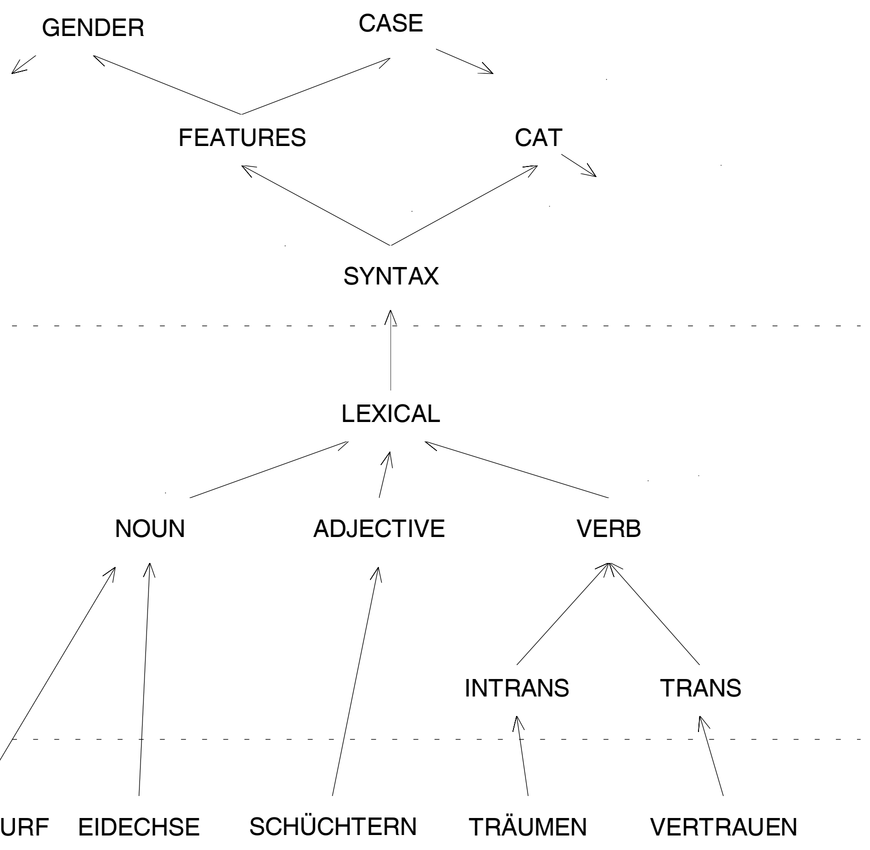
```
NODE1:    <path1> == av
          <path2> == <path1>.
```

By a convention DATR nodes are represented with capital letters.

[4] It is sometimes claimed that a special interface is required to combine a feature-based system and a DATR lexicon (cf. Krieger/Nerbonne 92). As is shown in Kilbury/Naerger/Renz 1991 this is not the case since the desired feature structures can be obtained directly using DATR alone.

This strategy permits us to conceive of the overall lexicon as having the modular architecture which is shown in figure (3).

(3) overall structure of the lexicon[5]



In figure (3) three different kinds of nodes can be distinguished. Nodes like **MAULWURF** ('mole'), **EIDECHSE** ('lizard'), or **SCHÜCHTERN** ('shy') correspond to lexical entries, nodes like **NOUN** or **ADJECTIVE** represent types, and nodes likes **SYNTAX** or **GENDER** define parts of the resulting feature structures. Every lexical node inherits information from exactly one type node since we assume that every lexical entry is associated with a unique lexical

[5] Note that this is a small fragment just for illustrating the general structure of our lexicon. Only a few features, categories, and lexemes are listed. The arrows indicate the flow of inheritance: 'A $\rightarrow$ B' means that A inherits from B.

type[6]. These parent nodes, which constitute a proper subset of all type nodes, are therefore called lexical type nodes. In this way we employ the concept of lexical types to partition lexemes into disjoint classes. This structure is not imposed by DATR itself; rather, DATR is used as a tool for representing our lexicon theory (for a similar approach cf. Gibbon 92 or Cahill/Evans 90).

Whereas the lexical nodes do not build a hierarchical structure, type nodes are hierarchically connected according to generality. More specific type nodes inherit information from more general ones. Like lexical nodes, every type node inherits from a single more general type so that a tree structure emerges. This means that the DATR facility of multiple inheritance is not used in defining the lexical and type nodes. The most general type node **LEXICAL** refers to the structure-building nodes, which in turn form a subnet which is also hierarchical but not tree-structured since multiple inheritance is necessary. The small downward arrows indicate that structure-building nodes may refer to lexical nodes in order to access the information specific to particular lexemes.

Type nodes and lexical nodes are very similar in structure. Information that is specific to a node is stated directly at the node with a path and an atomic value, potentially overriding the more general information. Information that the node has in common with other nodes is inherited from a single more general node with the empty path (<>). In figure (4) a lexical entry for the German word *Maulwurf* ('mole') is illustrated together with its lexical type **NOUN**.[7]

[6] In our example lexical nodes inherit only from type nodes, but this assumption only holds when phrasal lexemes are ignored. Collocations and idioms require lexical nodes to inherit from each other.

[7] In order to simplify the discussion, full forms rather than lexemes proper are represented here and throughout the rest of the paper. Of course, the marking of nouns for number is a matter of inflectional morphology, but we treat it as lexical information here.

(4) lexical entry with corresponding type

```
MAULWURF:       <> == NOUN
                <gender> == masculine
                <number> == singular.

NOUN:           <> == LEXICAL
                <cat> == n
                <status> == normal
                <person> == third.
```

As a noun, **MAULWURF** inherits all its general information like category and person from **NOUN** by default. On the other hand the information about gender and number which is specific to **MAULWURF** is defined directly at the node. The information for **NOUN** is structured analogously. This division into general and specific information is not accidental but intentional and linguistically desirable since it captures significant generalizations.

Besides their structural similarity, type nodes and lexical nodes share another property: queries consisting of the node and the empty path (i.e. **MAULWURF:<>**, **NOUN:<>**) can be evaluated, but the results differ. The values associated with the lexical nodes are fully-specified feature structures, while those associated with type nodes are underspecified feature structures in the sense that they include all features which every instance of this type exhibits, but some of these features lack a specific, lexically determined value. As will be shown later, this difference can be exploited in formulating new lexical entries for unknown words.

In order for new lexical entries to be integrated into the existing lexical net they must have the canonical structure common to all lexical entries. That means that they have a single path which refers to a corresponding type by default inheritance and several other paths which contain information specific to the particular unknown word.

The basis for this approach is our assumption that new lexical entries introduce no structural changes in the lexicon. While this assumption obviously would be inappropriate in the case of first language acquisition, it is plausible for the simulation of idealized lexical acquisition in adults, which is our concern here.

3. A general strategy for simulating the acquisition of new lexical entries

Our basic assumption about the acquisition of unknown words is that their analysis involves the same principles as the processing of natural language in general. This means in particular that no special component should be needed that is not independently required for parsing input with known words.

The parser adopted uses a left-corner algorithm with a top-down filter (cf. Wirén 87, Kilbury 90). The top-down filter takes the form of a transitive binary relation linking the categories of parsed constituents with those of current syntactic goals. Despite its designation as a "top-down" filter, the relation links information nondirectionally and can serve to answer different questions. If a lexical entry is found for the current input word, the relation says whether the entry can be useful for achieving the current hypothetical expectation. If we have no suitable entry, the relation can give us a lexical category that may be useful, depending on the further input. Instead of selecting an arbitrary lexical category for the input word when no entry is found, we therefore use the linking relation to propose a possible representation on the basis of the current expectation.

Consequently, expectations about the next syntactic constituent to be analyzed play a central role in the parsing process. Whereas the parsing algorithm itself remains unchanged, the lexical lookup is slightly modified so that if no appropriate entry for a word is found, the linking relation provides a candidate representation.

This representation is assumed to meet certain conditions which are defined in terms of the distinction between open and closed lexical classes[8]. On the one hand the proposed category must belong to an open lexical class, e.g. no new determiners or conjunctions are possible. On the other hand the unknown word generally may not be homonymous with a word belonging to a closed lexical class, e.g. the word form *das*, which already has an entry as a definite article, cannot be reinterpreted as a new German noun. The first condition constitutes an absolute constraint on unknown words since all items of closed classes are assumed

[8] Basically, all function words like determiners, conjunctions, or pronouns are viewed as belonging to closed lexical classes, which means that no new members are possible. In contrast, the classes of nouns, verbs, adverbs, and adjectives are considered to be generally open, although certain semantically based exceptions arise (e.g. the class of lexemes designating directions: *east*, *west*, *north*...).

to be known. In contrast, the second condition must be regarded as a heuristic since it allows exceptions (note e.g. the English modal verb forms *can* and *might* and the corresponding homonymous nouns).

Furthermore, it is assumed that every property of an unknown word is regular. These properties concern all linguistic levels. For example, new German verbs always subcategorize for a normal nominative subject (i.e. no new expletive verbs are possible), and inflected word forms belong to the regular morphological paradigms. Thus the concept of an open lexical class is further restricted by these assumptions.

Since we presuppose the unification paradigm as exemplified by PATR-II (cf. Shieber 86), strings are combined by concatenation, feature structures are the only data type, and unification is the sole operation on feature structures. Consequently, the expectation for an unknown word must be represented by a partially specified feature structure. During the analysis of a sentence further information about an unknown word may become available from the syntactic and semantic context. This information is monotonically combined with the representation of the unknown word through unification.

The resulting feature structure should contain all and - most importantly - only the relevant information about the unknown word. The next section discusses how this is achie-ved.

4. Collecting the relevant information for unknown words

The formulation of new lexical entries requires that all relevant information about the un-known words be available. As mentioned above, within the unification paradigm feature structures constitute the sole information-bearing data structure. This means that the linguistic information about the unknown word must be collected by unification in attribute-value pairs with either atomic or complex values.

In general, information about an unknown word is available from the given linguistic context, i.e. the rules applied in an analysis (context-free skeleton and corresponding unifica-tion part) and all other constituents. The flow of information is illustrated in the following toy grammar and the analysis of the sentence *die Nelfe schläft* ('the ??? sleeps')[9], where *Nelfe* is assumed to be unknown.

(5) rules[10]

1. *S* → *NP IV*

$$\begin{bmatrix} \mathit{S}: & [\mathit{cat}:\ \mathit{s}] \\ \mathit{IV}: & \begin{bmatrix} \mathit{cat}: & \mathit{iv} \\ \mathit{subcat}: & \begin{bmatrix} \mathit{first}: & *1* \\ \mathit{rest}: & \mathit{end} \end{bmatrix} \end{bmatrix} \\ \mathit{NP}: & *1*[\mathit{cat}:\ \mathit{np}] \end{bmatrix}$$

2. *NP* → *DET N*

$$\begin{bmatrix} \mathit{NP}: & \begin{bmatrix} \mathit{cat}: & \mathit{np} \\ \mathit{agr}: & *1* \end{bmatrix} \\ \mathit{DET}: & \begin{bmatrix} \mathit{cat}: & \mathit{det} \\ \mathit{agr}: & *1* \end{bmatrix} \\ \mathit{N}: & \begin{bmatrix} \mathit{cat}: & \mathit{n} \\ \mathit{agr}: & *1* \end{bmatrix} \end{bmatrix}$$

[9] Since *Nelfe* is an invented form, we provide no English translation.

[10] Instead of equations as in Shieber 86 we use the matrix notation for representing the unification part of a context-free rule. Reentrancies are represented by **N**, where *N* is an integer. The names for particular feature structures appear in these matrices as attributes.

(6) lexicon

die[11]

$$\begin{bmatrix} \textit{cat:} & \textit{det} \\ \textit{agr:} & \begin{bmatrix} \textit{gender:} & \textit{feminine} \\ \textit{number:} & \textit{singular} \\ \textit{case:} & \textit{accusative} \vee \textit{nominative} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textit{cat:} & \textit{det} \\ \textit{agr:} & [\textit{number:} \ \textit{plural}] \end{bmatrix}$$

schläft

$$\begin{bmatrix} \textit{cat:} & \textit{iv} \\ \textit{subcat:} & \begin{bmatrix} \textit{first:} & \begin{bmatrix} \textit{cat:} & \textit{np} \\ \textit{agr:} & \begin{bmatrix} \textit{case:} & \textit{nominative} \\ \textit{number:} & \textit{singular} \\ \textit{person:} & \textit{third} \end{bmatrix} \end{bmatrix} \\ \textit{rest:} & \textit{end} \end{bmatrix} \end{bmatrix}$$

Schläft is an intransitive verb (***iv***) which subcategorizes for a noun phrase (***np***) with fixed features (***case***, ***person***, ***number***). When rule 1 is applied in the analysis of the example sentence, the expectation NP is assigned to the phrase *die Nelfe* and the reentrancy ***1*** unifies the subcategorization information with the expected NP. This produces the following feature structure for the noun phrase:

[11] In this lexical entry disjunction is represented with $\vee$. For the analysis presented, the form of the representation is not important. Multiple entries as well as other notations would lead to the same results.

(7) feature structure for the noun phrase *die Nelfe*

$$\begin{bmatrix} \textit{cat:} & \textit{np} \\ \textit{agr:} & \begin{bmatrix} \textit{case:} & \textit{nominative} \\ \textit{number:} & \textit{singular} \\ \textit{person:} & \textit{third} \end{bmatrix} \end{bmatrix}$$

In order to parse the noun phrase *die Nelfe*, rule 2 is applied. Since *die* is a determiner the expectation N is assigned to the unknown word *Nelfe*. The reentrancy between ***np*** and ***det*** ensures that the right information about the determiner is selected (the specifications ***gender: feminine***, ***number: singular***, ***case: nominative***) and unifies the ***agr***-features. The resulting feature structure for the unknown word *Nelfe* is the following:

(8) feature structure for the unknown noun *Nelfe*

$$\begin{bmatrix} \textit{cat:} & \textit{n} \\ \textit{agr:} & \begin{bmatrix} \textit{case:} & \textit{nominative} \\ \textit{number:} & \textit{singular} \\ \textit{person:} & \textit{third} \\ \textit{gender:} & \textit{feminine} \end{bmatrix} \end{bmatrix}$$

This feature structure contains exactly the information desired. Some of the information comes from the determiner (***gender***, ***number***, ***case***) and some from the verb (***case***, ***number***, ***person***). Whereas in this example all specifications are appropriate in the sense that they are lexically relevant for nouns, problems arise when information is collected that is not relevant for the unknown word. If, for example, determiners had an additional ***agr***-feature for definiteness, this information would also be present in the feature structure for *Nelfe* although, with the exception of mass nouns, German nouns are not lexically specified for the feature definiteness. This problem seems to have gone unnoticed by most investigators with the exception of Lytinen/-Roberts 89. In their paper on lexical acquisition they analyze the noun phrase *a sandwich*, where *sandwich* is assumed to be unknown. Because *a* is specified as the indefinite article, *sandwich* is specified as indefinite in the resulting feature structure. The authors observe (footnote 3) "We know that, in general, 'sandwich' does not always refer to only objects whose reference is indefinite, because this information is in the lexical entry of 'a'", but they do not propose a solution.

The problem described here is not restricted to a particular feature but arises frequently in unification-based grammars. For example, if a German sentence is parsed in which the verb is unknown, the feature structure assigned to the verb will contain inappropriate, contextual information about the latter's complements. The feature structure of its subject will not only bear information about case and number but also about gender and definiteness. For objects, only case information plays a role in the lexical subcategorization of the verb, and all other information specific to German noun phrases (like gender, number, and definiteness) is extraneous.

The reason for this overspecification is that, typically, whole chunks of information (like the values of ***agr*** or ***subcat***) are unified. The problem arises regardless of whether reentrancies in rules (as in the toy grammar above) or general principles (like the 'subcategorization principle' in HPSG, Pollard/Sag 87) unify these chunks. The result of the unification is normally appropriate for the analyzed phrases, since the resulting feature structures are examined and the grammar is revised if the results are not the desired ones. The feature structures for smaller parts of these phrases (especially words), however, can be highly overspecified since unification is nondirectional. As a consequence, information which is appropriate for larger constituents can also be passed on through unification to smaller constituents where this information is not adequate. Since the feature structures of these constituents are normally not examined, the overspecification is not noticed.

The problem becomes apparent when sentences with unknown words are parsed and the resulting feature structures for these words are examined. Then the extraneous information is immediately recognized. Nevertheless, the issue should not be restricted to the area of unknown words but rather be viewed in the context of unification grammar and grammatical theory in general. The feature structures associated with known words in a particular analysis are overspecified in the same way, but this usually is overlooked. In this respect linguists and computational linguists have not fully recognized the fact that linguistically adequate structural descriptions must be correct not only for entire phrases but also for all of their constituents.

In our view a grammar can characterize a language adequately only when this latter requirement is met.

Two general strategies for solving the problem of overspecified representations for unknown words can be distinguished. The first utilizes a filter, that is, a mechanism which operates on the overspecified representations for unknown words. Such a mechanism isolates the relevant (i.e. lexically inherent) information from the feature structure, thereby producing an adequate representation for the unknown word. Although correct feature structures are finally achieved with this approach, it does not constitute a satisfactory solution for the problem since it fails to provide an adequate description of the distribution of information in the analysis of complex forms.

A satisfactory and general solution must apply to known and unknown words in the same way. Achieving this requires a different and more careful formulation of the grammar. The transportation of information should be fully motivated and linguistically justified in the description of each syntactic constituent. This means that, on the one hand, the appropriate features for each constituent must be specified. On the other hand, rules or principles must define which constituents share which features. An adequate grammar in accord with these considerations can be formulated which meets the linguistic requirement that every constituent be associated with a feature structure representing all and only the information appropriate to it.

For a particular grammar fragment the above considerations have certain consequences for our notion of "bundled" information, i.e. specifications that appear together within the complex value of an attribute. Whether or not a given feature is linguistically relevant and therefore specified in the formulation of the feature structure for some constituent depends on general considerations involving, among other things, a linguistic theory of agreement and government, which we will not discuss here. Features can and should be bundled when the following two conditions are simultaneously met:

- They always occur together, i.e. for every constituent of a grammatical phrase, if one feature is specified in the corresponding feature structure, then the other(s) is/are specified, too.
- They covary such that if the feature structures of two constituents *A* and *B* have the same specification for the feature *f*, they also have the same specification for the feature *g*.

Example:

In German, subjects agree with the verb in number and person, but not with regard to gender. Thus, ***number*** and ***person*** can be bundled as ***agr*** features, whereas ***gender*** cannot be included in this bundle.

In figure (9) the bundling of features according to these conditions is illustrated with a simplified feature structure for the noun phrase *ein Maulwurf* ('a mole').

(9) feature structure for the noun phrase *ein Maulwurf*

$$\begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{head}: & \begin{bmatrix} \textit{major}: & \begin{bmatrix} \textit{n}: & \textit{yes} \\ \textit{v}: & \textit{no} \end{bmatrix} \\ \textit{case}: & \textit{nominative} \\ \textit{status}: & \textit{normal} \end{bmatrix} \\ \textit{cat}: & \textit{np} \\ \textit{agr}: & \begin{bmatrix} \textit{number}: & \textit{singular} \\ \textit{person}: & \textit{third} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

In figure (9) the three bundles of features appearing as the values of ***head***, ***cat***, and ***agr*** are distinguished. The ***head*** features are all those for which all nominal projections are specified. Furthermore, all noun phrases are restricted with respect to these features in their syntactic roles as complements (subjects and objects). While the ***agr*** features are also specified for all nominal projections and thus for all nominal complements, only subjects are restricted with respect to these features since German objects do not agree with verbs in number and person. Therefore they should not be bundled together with the ***head*** features.

This arrangement is necessary for the verbal subcategorization so that objects are specified only for their ***head*** features and subjects for their ***head*** and ***agr*** features. Figure (10) shows a representation of the subcategorization information of the transitive verb *vertraut* ('trust').

(10) subcategorization information of the transitive verb *vertraut*

$$\begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{subcat}: & \begin{bmatrix} \textit{first}: & \begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{head}: & \begin{bmatrix} \textit{major}: & \begin{bmatrix} \textit{n}: & \textit{yes} \\ \textit{v}: & \textit{no} \end{bmatrix} \\ \textit{case}: & \textit{nominative} \\ \textit{status}: & \textit{normal} \end{bmatrix} \\ \textit{agr}: & \begin{bmatrix} \textit{number}: & \textit{singular} \\ \textit{person}: & \textit{third} \end{bmatrix} \\ \textit{cat}: & \textit{np} \end{bmatrix} \end{bmatrix} \\ \textit{rest}: & \begin{bmatrix} \textit{first}: & \begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{head}: & \begin{bmatrix} \textit{major}: & \begin{bmatrix} \textit{n}: & \textit{yes} \\ \textit{v}: & \textit{no} \end{bmatrix} \\ \textit{case}: & \textit{dative} \\ \textit{status}: & \textit{normal} \end{bmatrix} \\ \textit{cat}: & \textit{np} \end{bmatrix} \end{bmatrix} \\ \textit{rest}: & \textit{end} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

This represention bundles the information in a way that allows the formulation of linguistically justified rules or principles. These rules define the distribution of information in such a way that the representation of every constituent contains only its appropriate features. How the rules are formulated depends on which constituents are to be combined. In the case of subject-verb combinations the ***head*** and ***agr*** features have to be unified, while in the case of all other complements only the ***head*** features are shared. If the toy grammar given above in figures (5) and (6) is modified according to these considerations, rule 1 appears as in figure (11).

(11)

1. $S \rightarrow NP \quad IV$

$$\left[\begin{array}{ll}
S: & [syn: \ [cat: \ s]] \\
IV: & \left[\begin{array}{ll} syn: & \left[\begin{array}{ll} cat: & iv \\ subcat: & \left[\begin{array}{ll} syn: & \left[\begin{array}{ll} first: & \left[\begin{array}{ll} head: & *1* \\ agr: & *2* \\ cat: & np \end{array}\right] \\ rest: & end \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\
NP: & \left[\begin{array}{ll} syn: & \left[\begin{array}{ll} cat: & np \\ head: & *1* \\ agr: & *2* \end{array}\right] \end{array}\right]
\end{array}\right]$$

The main property of this description that makes it linguistically more adequate lies in the fact that all constituents are assigned only the information that is truly relevant for them. Since unknown words also appear as constituents, they are correctly specified, and overspecification never arises. Thus, this approach is general and alleviates the need for a special device like a filter to extract the lexical information from feature structures associated with word forms in a context.

5. Formulation of new lexical entries

After parsing is completed the incrementally constructed feature structure for an unknown word contains all the information that is appropriate in the sense discussed above. Nevertheless, this feature structure (see figure (12) with an example for a hypothetical unknown noun *Nolf* in the sentence *das Nolf träumt* - 'the ??? dreams') is not a lexical representation since it does not have the corresponding form discussed above in section 2. Furthermore, new lexical entries have to be integrated into the existing lexical net and related to the lexical types within it.

(12) feature structure for the unknown noun *Nolf*

$$\begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{head}: & \begin{bmatrix} \textit{major}: & \begin{bmatrix} \textit{n}: & \textit{yes} \\ \textit{v}: & \textit{no} \end{bmatrix} \\ \textit{case}: & \textit{nominative} \\ \textit{status}: & \textit{normal} \end{bmatrix} \\ \textit{cat}: & \textit{n} \\ \textit{agr}: & \begin{bmatrix} \textit{number}: & \textit{singular} \\ \textit{person}: & \textit{third} \end{bmatrix} \\ \textit{gender}: & \textit{neuter} \end{bmatrix} \end{bmatrix}$$

Our proposal requires no special devices in order to parse phrases containing unknown words or to determine the relevant information of the latter. In sections 3 and 4 it was shown that our approach allows a uniform treatment for both known and unknown words. But in the special case of unknown words the additional problem of formulating new lexical entries on the basis of feature structures arises, which necessitates a further component for this particular task.

Our aim is to develop a general device which operates on form (the feature structures) but not on content (the represented linguistic information) to uniformly transform any feature structure associated with a word form into a corresponding lexical entry. Thus, the component should constitute a kind of compiler which can be defined in the form of an algorithm.

Whereas the algorithm should be independent of the grammar, it has to be adapted to the particular notation chosen for lexical entries. In our approach a lexical entry encoded in DATR consists of a lexical node with an empty path referring to the lexical type and some nonempty paths (with atomic values) stating the information specific to the lexeme.

The strategy for building the lexical entry consists of three main steps:

(1) The lexical type of the unknown word is determined.

(2) The information specific to the word is found by comparing its full feature structure with the less specific feature structure obtained by evaluating the word's lexical type.

(3) The information identified in (2) is formulated with the corresponding DATR paths-value pairs.

In the following, these steps are discussed in more detail.

Identification of the lexical type for an unknown word is based on the assumption that the class of possible candidates for the type is limited. Corresponding to the concept of open and closed lexical classes (cf. section 3 above) we define a class of open lexical types, which includes for example nouns, transitive, and intransitive verbs, but not determiners or expletive verbs. These open lexical types form a proper subset of the lexical types in general, which are a proper subset of all types as mentioned in section 2. We further assume that the class of open lexical types partitions the set of possible new words, so that every unknown word in a given context corresponds to exactly one of these open lexical types. A possible approach for determining the type of the unknown word consists in evaluating the open types successively until the type is found. The evaluation of a type results in an underspecified feature structure which represents the information common to all lexical entries of this type (see figure (13) for the lexical type **NOUN**). The feature structure of the type is then compared with the feature structure of the unknown word, and if it subsumes the latter, it is the sought type.

(13) feature structure for lexical type NOUN[12]

$$\left[\textit{syn}: \left[\begin{array}{ll} \textit{head}: & \left[\begin{array}{ll} \textit{major}: & \left[\begin{array}{ll} \textit{n}: & \textit{yes} \\ \textit{v}: & \textit{no}\end{array}\right] \\ \textit{status}: & \textit{normal} \\ \textit{case}: & [\,] \end{array}\right] \\ \textit{cat}: & \textit{n} \\ \textit{agr}: & \left[\begin{array}{ll} \textit{person}: & \textit{third} \\ \textit{number}: & [\,]\end{array}\right] \\ \textit{gender}: & [\,] \end{array}\right]\right]$$

The strategy just sketched is not goal-directed because the order in which the open lexical types are checked is not determined by the feature structure of the new word. An alternative,

[12] *[]* represents the maximally underspecified value of an attribute. In figure (13) ***gender:[]*** means that nouns are always specified for gender, but the particular value depends on the lexeme.

goal-directed search strategy can be introduced that takes the given feature structure and the tree of type nodes illustrated above in figure (3) as input. The search begins at the most general type node LEXICAL and descends recursively. We assume that a given type node has at most one daughter whose corresponding feature structure subsumes the feature structure of the unknown word. The recursion terminates when the most specific open lexical type node is found whose feature structure subsumes that of the unknown word.

After the type is determined, its feature structure is compared with that of the unknown word. A third feature structure is constructed that subsumes that for the unknown word but shares no specifications with that for the type; i.e. it expresses the specifically lexical information of the new entry.

The algebraic relationships that hold between the three feature structures (the feature structure of the unknown word **UW**, the feature structure of the lexical type **LT**, and the feature structure that is specific to the unknown word **S**) are stated in (14). The symbols $\sqcup$, $\sqcap$, $\sqsubset$, and $\setminus$ denote unification, generalization, subsumption, and difference, respectively.

(14) algebraic relationships between S, LT, and UW

$$S \sqcup LT = UW$$

$$S \sqcap LT = [\ ]$$

$$S = UW \setminus LT \quad \text{where } LT \sqsubset UW$$

An algorithm based on the relationships in (14) that constructs **S** from **UW** and **LT** is sketched in figure (15).

(15) algorithm for isolating the information specific to an unknown word

```
(0)  Does UW contain a(nother) path?[13]
     (1) no:    end
     (2) yes:   extract one path P_1 of UW
                (3)    determine the value V_1 of P_1
                (4)    V_1 = [] ?
                       (5) yes:  goto (0)
                       (6) no:   extract P_1 from LT
                                 (7)    determine the value V_2 of P_1
                                        (8)    V_2 = [] ?
                                               (9) no:   goto (0)
                                               (10) yes: collect P_1 and V_1
                                                         (11) goto (0)
```

Stated informally, this algorithm collects all paths and their values from the feature structure of the unknown word where the corresponding paths in the type feature structure have less specific values. The case that the type feature structure has no corresponding path at all cannot arise because of our notion of types. The paths thus obtained represent all and only the information that is specific to the unknown word. When this algorithm is applied to the feature structure of the hypothetical unknown noun *Nolf* (see figure (12)) and its corresponding type **NOUN** (see figure (13)), the following feature structure emerges containing the information specific to *Nolf*.

[13] In this context a path is considered a complete sequence from the root to an atomic value or variable.

(16) information specific to the unknown noun *Nolf*

$$\begin{bmatrix} \textit{syn}: & \begin{bmatrix} \textit{head}: & [\textit{case}: \ \textit{nominative}] \\ \textit{agr}: & [\textit{number}: \ \textit{singular}] \\ \textit{gender}: & \textit{neuter} \end{bmatrix} \end{bmatrix}$$

In order to obtain an appropriate lexical entry the PATR paths in the above feature structure must be transformed into corresponding path-value pairs in DATR. Since our grammar and the lexicon are attuned to each other, there exists a strong correspondence between PATR paths and DATR path-value pairs. In general, the last attribute-value pair of a PATR path matches one path-value pair in DATR. Thus, the above feature structure (figure (16)) corresponds to the three path-value pairs in figure (17).

(17) path-value pairs specific to the unknown word *Nolf*

```
<case> == nominative
<number> == singular
<gender> == neuter
```

The relationship between PATR paths and DATR path-value pairs is different when complements are involved; this is the case in the feature structure for verbs (see figure (10), which illustrates the subcategorization of a verb). Here not only the last PATR attribute-value pair but the whole path is important because of the way the list of complements is encoded with **first** and **rest**. In the feature structure of figure (10), all information about the first complement (i.e. the subject) is represented as the complex value of the path ***syn|subcat|first***[14]; the information about the second complement (i.e. the indirect object) appears as the value of the path ***syn|subcat|rest|first***. In our lexicon, information about complements is stated in DATR paths which contain an identifier (i.e. a name) for the complement (e.g. comp2) in addition to the particular attribute (e.g. case). Naturally, this information is also divided into general

[14] To avoid confusion with the notation for DATR paths we use the HPSG notation (Pollard/Sag 87) for paths here.

information about the type and that specific to the lexeme. Thus, the fact that the object of the German verb form *vertraut* ('trusts') takes the dative case is stated directly in the lexical entry with the DATR path-value pair **<case comp2> == dative**, whereas information about the category (noun phrase) of the object is inherited from the verb's type.

Now a new lexical entry for the unknown word can be constructed. First the orthographic representation of the word given in capital letters is taken as the node name of the new lexical entry. The specific path-value pairs discussed in the last paragraph are then associated with this node. Finally, a path-value pair consisting of the empty path and the reference to the type node is added. For our example *Nolf* the new lexical entry would appear as in (18).

(18) new lexical entry for *Nolf*

```
NOLF:     <> == NOUN
          <gender> == neuter
          <case> == nominative
          <number> == singular.
```

This new lexical entry has exactly the same structure as that chosen for entries of all other nouns in the lexicon. Likewise, new adjectives, adverbs, and verbs are assigned their typical structure. These acquired lexical entries can be integrated into the existing lexicon simply by adding them. They are related to the overall lexical net by their reference to one of the existing lexical types.

6. Conclusion

The technique we have presented for creating new lexical entries relies on two main prerequisites. The first is our theory of the lexicon, which includes the central concept of lexical types in a hierarchical net. These lexical types allow us to distinguish between information specific to a lexeme and general information which is inherited from a single lexical type (a linguistic generalization which is crucial for the construction of new lexical entries). The second prerequisite involves new considerations about grammars in unification-based systems. These considerations make it possible to formulate a grammar which defines all and only the relevant information for each constituent, be it known or unknown.

The notions of lexical types and well-defined representations for unknown words form the basis for our straightforward algorithm that formulates new lexical entries. Although its strategy is general, it is not completely independent of the grammar. Transformation of the PATR paths into corresponding DATR sentences requires a correspondence between the statements in the DATR lexicon and the attribute-value pairs of the PATR feature structure. Our main goal of simulating the acquisition of new lexical entries for unknown words is achieved with the algorithm because it generates new lexical entries with the same structure as those already in the lexicon and integrates them therein.

Bibliography

Bouma, Gosse (1990) Defaults in Unification Grammar. In *Proc. of the 28th Conference of the ACL*, 165-171.

Cahill, Lynne J. / Evans, Roger (1990) An Application of DATR: the TIC Lexicon. In *Proceedings of the ECAI-90*, 120-125.

Evans, Roger / Gazdar, Gerald (1989a) Inference in DATR. In *Proc. of the 4th Conference of the EACL*, 66-71.

Evans, Roger / Gazdar, Gerald (1989b) The Semantics of DATR. In A. Cohn (ed.) *AISB89, Proc. of the 7th Conference of the Society for the Study of Artificial Intelligence and Simulation of Behaviour*, 79-87. London: Pitman.

Evans, Roger / Gazdar, Gerald (eds.) (1990) *The DATR Papers: February 1990 (= Cognitive Science Research Paper 139)*. School of Cognitive and Computing Sciences, University of Sussex, Brighton, England.

Flickinger, Daniel / Pollard, Carl / Wasow, Thomas (1985) Structure-Sharing in Lexical Representation. In *Proc. of the 23th Conference of the ACL*, 262-267.

Gibbon, Dafydd (1992) ILEX: Linguistic Approach to Computational Lexica. In Ursula Klenk (ed.) *Computatio Linguae*. Beihefte der Zeitschrift für Dialektologie und Linguistik, 32-53.

Kilbury, James (1990) QPATR and Constraint Threading. In *Proc. of 13th COLING*, 382-384.

Kilbury, James / Naerger, Petra / Renz, Ingrid (1991) DATR as a Lexical Component for PATR. In *Proc. of the 5th Conference of the EACL*, 137-142.

Krieger, Hans-Ulrich / Nerbonne, John (1992) Feature-Based Inheritance Networks for Computational Lexicons. In T. Briscoe / A. Copestake / V. de Paiva (eds.) *Default Inheritance within Unification-Based Approaches to the Lexicon*.

Lytinen, Steven L. / Roberts, Susan N. (1989) Lexical Acquisition as a By-Product of Natural Language Processing. In U. Zernik (ed.) *Proc. of the First International Lexical Acquisition Workshop*. Detroit, Michigan.

Pollard, Carl / Sag, Ivan (1987) *Information-Based Syntax and Semantics: Vol. 1 - Fundamentals*. Stanford, Calif.:CSLI.

Shieber, Stuart M. (1986) *An Introduction to Unification-Based Approaches to Grammar*. Stanford, Calif.:CSLI.

Wirén, Mats (1987) A Comparison of Rule-Invocation Strategies in Context-Free Chart Parsing. In *Proc. of the 3rd Conference of the EACL*, 226-233.

Zernik, Uri (ed.) (1989) *Proc. of the First International Lexical Acquisition Workshop*. Detroit, Michigan.