Multiword expressions : linguistic precision and reusability

Copestake, Ann; Lambeau, Fabre; Villavicencio, Aline; Sag, Ivan; Bond, Francis; Baldwin, Timothy; Flickinger, Dan

2002

Copestake, A., Lambeau, F., Villavicencio, A., Bond, F., Baldwin, T., Sag, I., et al. (2002). Multiword expressions : linguistic precision and reusability. In proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002) pp.1-7.

https://hdl.handle.net/10356/90700

Multiword expressions: linguistic precision and reusability

Ann Copestake*, Fabre Lambeau*, Aline Villavicencio*, Francis Bond‡,
Timothy Baldwin†, Ivan A. Sag†, Dan Flickinger†

* University of Cambridge Computer Laboratory,
William Gates Building, JJ Thomson Avenue,
Cambridge, CB3 0FD, UK
{aac10,faml2,av208}@cl.cam.ac.uk
‡ NTT Communication Science Laboratories,
Hikari-dai, Seika-cho, Soraku-gun, Kyoto 619-0237, JAPAN
bond@cslab.kecl.ntt.co.jp
†CSLI, Ventura Hall, Stanford University
Stanford, CA 94305-4115, USA
{tbaldwin,sag,danf}@csli.stanford.edu

Abstract

This paper discusses the approach to multiword expressions being adopted in the LinGO English Resource Grammar (http://lingo.stanford.edu), a broad-scale bidirectional grammar of English in the HPSG framework. We discuss how the lexicon of multiword expressions is encoded in a database and describe the implications for building a reusable lexical resource.

1. Introduction

Multiword expressions (MWEs) are generally acknowledged to be a serious problem for many areas of language technology. MWEs include not only nominal compounds, phrasal verbs, idioms and collocations, but also less easily classified examples: e.g., *by and large*, *ad hoc*, *in line*, *for instance*, *such as*, *and so on*. Some MWEs are completely fixed, such as *ad hoc*, but others allow differing amounts of variability. This is related to decomposability: i.e., whether meaning can be assigned to the parts of the MWE (Nunberg et al., 1994). Another factor is productivity: some combinations of verbs and particles are idiosyncratic (e.g., *look up* in the *refer to* sense) while others are somewhat predictable (e.g., *eat up*, *drink up*).

Coverage of MWEs in existing lexical resources is uneven. For instance, the Alvey Tools lexicon and COMLEX have a good coverage of the syntactic properties of phrasal verbs, but do not indicate whether the entries can be regarded as compositional or productively formed. Neither resource covers idioms, or verbs with fixed adverbs, such as *set aside* or *go overboard*. WordNet has a large number of MWEs, but does not describe their variability or handle idioms: handling idioms in WordNet is not straightforward (Fellbaum, 1998). Some attempts have been made to develop a standard for encoding MWEs (especially within EAGLES), and the XMELLT project is attempting to lay the groundwork for large scale encoding of MWEs, but so far no general standard exists.

Unfortunately there is no agreement within linguistics about the treatment of most classes of MWEs. In NLP, whether an MWE entry is required may depend on the depth of processing being undertaken: e.g., syntactically regular idioms are not very relevant if the system's output is a syntactic tree. It also depends on the grammar: a MWE such as *in line* could be treated by a rule that allows preposition-noun combinations for all nouns, but at the cost of overgeneration.

Contrast this situation with syntax: different theories have very different ways of lexically encoding basic syntactic categories, but there is broad agreement about the classes to be encoded. For the syntax and formal/compositional semantics of simple words, the LinGO English Resource Grammar (ERG) has a lexical database structure which essentially just encodes a triple: orthography, type, semantic predicate. Here 'type' is a single identifier whose interpretation may vary from grammar to grammar: the constraints on the type are expressed as typed feature structures (TFSs), but these constraints are regarded as part of the main grammar rather than the lexicon. The lexical entries are expanded into large TFS data structures when they are used in processing. It is therefore possible to develop and test a precise lexicon for one grammar and to reuse the open-class words it contains with a different framework: such interconvertibility has been practically demonstrated in several projects.

For example, see Table 1, which reflects the current LinGO ERG lexical database.[1] **v_np_trans_le** is the type for simple transitive verb lexical entries ('le' stands for lexical entry), **like_rel** is the semantic relation (which is also a type), and `like_v1` is the lexical identifier, which acts as the key for the database entry. We will follow the convention of using bold font for types and 'tt' font for identifiers in the text throughout this paper.

However, except for the simplest examples (such as *ad hoc*), which can be treated as words with spaces, MWEs

[1] This table excludes many database columns that are used for bookkeeping purposes (such as date and source of entry), and some columns needed for MWEs, to be discussed later. The current database was initially automatically generated from the original form of the ERG lexicon, which was a simple text file containing the lexical entries expressed in the standard ERG description language, TDL. The core ERG lexicon was manually constructed — it can be augmented by entries automatically derived from COMLEX and other sources.

LEXEME_ID	ORTH	TYPE	SEMPRED
like_v1	like	v_np_trans_le	like_rel

Table 1: Simple lexical database entry

require more complex entries, which refer to multiple components of the structure. This is a problem even when developing a single grammar, since changes may invalidate the MWE lexicon. But our goal is the more ambitious one of building a resource which can be used in multiple frameworks.

We are addressing these issues by developing a typology of MWEs covering all the major classes, which is formally described and practically implemented (see http://lingo.stanford.edu/mwe/ for more information about the LinGO MWE project). Our approach is to use the LinGO ERG both as a tool for empirical investigation of MWEs and as a consumer (and therefore validator) of the resource. A MWE is postulated if and only if standard grammar rules and simplex entries do not suffice when we attempt to process some corpus data. Since the grammar is bidirectional, this prevents us from using shortcuts, such as the preposition-noun combination rule mentioned earlier, which would result in overgeneration. Naturally, the encodings of MWEs must be precise. However, this does not necessarily mean that we require complex lexical entries, which would limit reusability, because we can generalize over classes of MWEs. The goal of the current paper is to discuss how the lexicon of multiword expressions can be encoded in a database and describe the implications for building a reusable lexical resource.

The ERG can be used for parsing by a number of systems, but the discussion in this paper is centred around the LKB system (Copestake, 2002) which is the basis for our experiments with MWEs. The LKB can be used for both parsing and generation. The lexical database makes use of standard relational database technology and is currently running under Postgres, MySQL and Microsoft Access. We have chosen to use a relational database rather than an object-oriented design for two main reasons. The first is that we want to maintain a division between the work of the typed feature structure formalism and the database. The typed feature structure formalism supports inheritance and has many object-oriented properties: the database has to store the minimal amount of information necessary to construct TFSs, rather than to compete in functionality. Secondly, the LKB is freely distributed as open source and runs on multiple platforms: we want the lexical database to be usable as widely as possible.

In the rest of this paper, we discuss some specific classes of MWE, describing approaches to database representation, and showing how these relate to the representation used in the LinGO ERG up to now, and to the alternatives we are developing in the current MWE project.

2. Verb-particle constructions

The current LinGO ERG has entries for verb-particle constructions such as the entry for the verb component of *look up* (in the ‘refer to’ sense) shown in Figure 1 (in the TDL description language). This expands out to a typed feature structure, some parts of which are shown in Figure 2. The additional information in Figure 2 arises from the constraint on the type **v_particle_np_le**. Notice that the lexical entry uses the –COMPKEY feature for convenience (– is a conventional notation in the ERG for features with no linguistic significance). COMPKEY is coindexed with a path which goes deeply into the TFS, specifically the KEY value of the verb’s first complement. KEY values of lexical entries are systematically identified with the main lexical relation that occurs in the CONT(ENT). This TFS illustrates the main mechanism for variable MWEs in the ERG up to now: the entry illustrated only covers *look* but obligatorily selects for *up* as a complement, via its KEY relation (**_up_rel**).

The TFS shown corresponds to an example such as (1).

(1) Kim looked up the word

A lexical rule is used to construct the alternative TFS required for *Kim looked the word up*, where *up* is treated as the second complement: the application of this rule is controlled by the type of the lexical entry.

Selection via KEY is an alternative to the selection of prepositions via a feature such as FORM which was used in earlier versions of HPSG. It has the advantage of avoiding redundancy: the semantics for an entry has to be specified anyway, so exploiting it for selection purposes avoids the requirement for additional information. An entry of any part-of-speech can be selected via the KEY feature, whereas FORM was only used for a limited range of selection purposes.

The database entry that we currently use to represent the information in the lexical entry is shown in Table 2. As in Table 1, we exclude the bookkeeping information and we also exclude some additional columns similar to COMPKEY which are needed for other classes of entry which we will not discuss here. The LEXEME_ID is the database key, as with the simplex entries. The database entry is expanded into the TDL expression or directly into the TFS, depending on the way the database is being used with the LKB system.

There are additional constraints on database entries for MWEs. In particular, the COMPKEY should correspond to the SEMPRED of a particle lexeme entry (ideally a unique particle lexeme). However this constraint is not currently enforced, so errors can creep in.

Compared to the lexical database entry for the simplex words, reuse of this entry is somewhat problematic. The difficulty is that the database entry is relatively specific to the encoding adopted, due to the use of COMPKEY, in particular. Although it might be possible to reuse this structure

```
look_up_v1 := v_particle_np_le &
  [ STEM < "look" >,
    SYNSEM.LOCAL.KEYS [ KEY _look_up_rel,
                        --COMPKEY _up_rel ] ].
```

Figure 1: Description of a lexical entry for the verb part of *look up*

$$\begin{bmatrix} \textbf{v\_particle\_np\_le} & \\ \text{STEM} & \langle \textbf{"look"} \rangle \\ \text{SYNSEM} & \begin{bmatrix} \textbf{particle\_np\_verb} & \\ \text{LOCAL} & \begin{bmatrix} \text{CAT} & \begin{bmatrix} \text{HEAD} & \textbf{verb} \\ \text{VAL} & \begin{bmatrix} \text{SUBJ} & \langle \begin{bmatrix}\text{LOCAL} & \begin{bmatrix}\text{CAT} & \textbf{nomp\_cat\_min}\end{bmatrix}\end{bmatrix} \rangle \\ \text{COMPS} & \langle \begin{bmatrix}\text{LOCAL} & \begin{bmatrix}\text{KEYS} & \begin{bmatrix}\text{KEY} & \boxed{2}\ \textbf{\_up\_rel}\end{bmatrix}\end{bmatrix}\end{bmatrix}, \ldots \rangle \end{bmatrix}\end{bmatrix}\end{bmatrix} \\ \text{CONT} & \begin{bmatrix}\text{LISZT} & \langle \boxed{1} \begin{bmatrix}\textbf{\_look\_up\_rel} & \\ \text{ARG1} & \boxed{x} \\ \text{ARG3} & \boxed{y}\end{bmatrix} \rangle\end{bmatrix} \\ \text{KEYS} & \begin{bmatrix}\text{KEY} & \boxed{1} \\ \text{–COMPKEY} & \boxed{2}\end{bmatrix} \end{bmatrix}\end{bmatrix}$$

Figure 2: Typed feature structure for the verb part of *look up*

LEXEME_ID	ORTH	COMPKEY	TYPE	SEMPRED
look_up_v1	look	_up_rel	v_particle_np_le	lookup_rel
tidy_up_v1	tidy	_up_rel	v_particle_np_le	tidyup_rel

Table 2: Current verb-particle database entries

within another formalism, reuse would involve finding an appropriate mapping of this concept.

There are some other problems with the approach to verb-particle constructions. Because the same lexical entry is used for the particle *up* and the preposition, a predicate corresponding to *up* appears in the semantics for sentences such as (1). It is marked as being selected-for, and can therefore be treated as semantically vacuous by post-processing, but this is somewhat inelegant and causes problems for generation, since the predicate has to be inserted prior to inputting the logical form to the generator. Uniformly treating particles as semantically empty does not allow for the systematic contribution that *up* makes in combination with some classes of verbs.

In the current LinGO ERG, verb-particles are always listed explicitly and have special semantic relations (i.e., they are all treated as non-compositional). This means that the semantics for *tidy* and *tidy up* is unrelated, which is incorrect, since this use of *up* can be analyzed compositionally with a completive meaning (compare *rip up*, *tear up* and so on). Even for the non-productively formed *look up*, it is at least arguable that the semantics should have some relationship to the semantics of a sense of *look*: compare, for instance, *look for*. Note that this is a more transparent verb meaning than is found in the intransitive sense of *look up* meaning 'improve' (although the *up* in that case is relatively transparent, via a conventional metaphor).

The fact that there is no formal relationship in the current ERG between the particle-taking form and the base-form implies that an approach to stochastic HPSG making use of lexical forms or semantic predicates must treat the particle construction entirely separately from the base form. This is probably undesirable for the relatively transparent verb-particle constructions, since it removes the possibility of backing off to the base form in the case of sparse data.

There are two other potential reasons to link verb-particles with the base verb, both of which apply regardless of the compositionality of the verb-particle, although they don't currently impact the ERG processing. The first reason is that inflectional morphology always follows the same pattern as the base verb: if the base verb is irregularly inflected, so is the verb-particle verb. This does not cause any problems for the ERG/LKB combination, since irregular morphology is specified by string match rather than being part of the base lexical entry, but could cause problems for other grammar/parser combinations. Since the database does not link the verb-particle with a base verb, it could not be used by any theory where irregular morphology was encoded on the base lexeme. The second issue is that register and dialect information is generally shared between the base verb and the verb-particle verb. For instance, *piss* and *piss off* are generally both perceived as informal and impolite. The current ERG does not utilize register and dialect information, though will be extended to do so in the future.

In outline, our revisions to the ERG's treatment of verb-particles in the MWE project are as follows:

- Particles are treated as lexically distinct from prepositions (though the revised TFS for a particle can be constructed via a lexical redundancy rule from a preposition). The particle contains a KEY, but this is not linked to the CONT for the non-compositional particles: they are treated as semantically vacuous. The parser and generator postulate particles only when a licensing verb-particle sign is present, so this does not lead to significant efficiency problems.

- Compositional verb-particle constructions are generated by lexical rule and given compositional semantics: compositional particles are not semantically vacuous. For the time being, the lexical rule is treated as a lexical redundancy rule, which has to be licensed specifically, though this will change if we find we can encode some verb-particles as fully productive (this requires a finer granularity of semantic specification than we are currently using).

We will not discuss this approach in detail here — more details are given in Villavicencio and Copestake, 2002). Our point in this paper is to discuss the database and the implications for reusability.

Revised database entries are shown in Table 3. In the entry for `look_up_v1`, the predicate, **lookup_rel**, indicates that the semantics is idiosyncratic. However, the entry can still access information from the ordinary lexical entry for `look_v1`. In contrast, `tidy_up_v1` is relatively compositional, in that the particle `up_completive_p2` conveys an aspectual meaning, which is also found in *gobble up* etc. However, listing is necessary because the possible combinations are not fully predictable (c.f., *phone/ring/call/*telephone up*). The lexical database can be regarded as validating lexical redundancy rule application. This revised database format requires that different relations be used for different classes of MWEs, but this leads to a greater transparency and also better error checking.

`look_v1, up_noncomp_p1, tidy_v1` and `up_completive_p2` are foreign keys in database terms (they have to be the value of LEXEME_ID in some other relation) and they have to belong to specific groups of lexical types. The links to the base forms enable the inheritance of dialect and register information, so this formulation is more extensible than the old one. Semantic selection is still supported, since the KEY predicates of the base forms can be accessed. However, systems which did not adopt this approach to selection in their grammar of MWEs could probably also utilize the database, and similarly, we could continue to make use of it if we modified the approach.

The database still supports the old grammar, on the assumption that the database-to-TDL code autogenerates relations such as **tidyup_rel** and that the orthography is derivable from the foreign keys. One benefit of the revised approach is that we can support different grammars more easily, and possibly different formalisms. In effect, we are treating the database as a device from which to generate a typed feature structure lexicon, rather than as a direct implementation of the TDL. A cost is that more information has to be supplied by the lexicographer, in that it is necessary to explicitly consider base forms and compositionality, but for compositional verb-particle constructions this leads to improvements in the semantic representation, removing the need for meaning postulates that would otherwise have to be supplied (or transfer rules in the case of an MT system). In any case, since our practise is to enter verb-particle constructions at the same time as a base verb is added, the lexical entries can be generated semi-automatically with the lexicographer asked to select between possibilities and validate auto-generated entries. The non-compositional entries do not, in fact, strictly require a verb base form (assuming dialect and register are not an issue), so a default 'dummy' base form could be supplied: this is required for those cases where a verb-particle construction is found without a corresponding verb. The revised approach also requires different database relations for different classes of MWE (e.g., compound nouns use different columns), but these are governed by the lexical type and are thus predictable automatically.

We have discussed verb-particle constructions in some detail, because they illustrate how improvements to the representation can also lead to a more transparent database entry. Our revised approach to database entries for MWEs in general (other than words with spaces) is that they should (as far as possible) consist of an identifier, a type, a semantic predicate (if not compositional), and the constituent simplex forms, where the nature of the simplex forms and their interrelationship is determined by the type of MWE.

3. Compound nouns

We will only consider here English compound nouns that are spelled as distinct lexicographic words: we treat single word compounds as simplex words. Currently we ignore hyphens: this is not completely satisfactory, but we will not discuss this further here.

Example TDL entries for two compound nouns in the current ERG are shown in Figure 3. These compounds are treated as words with spaces. The current ERG only lists compounds in cases where the combination of nouns cannot be produced by the productive noun-noun compounding rule. In the examples in Figure 3, listing is required for *Easter Sunday* because holiday-denoting terms are of a different type from other nouns, and for *Menlo Park* because it is a proper name (and because *Menlo* does not exist as an independent word). Otherwise, even clearly lexicalized compounds such as *car park* are treated as being compositionally formed. For instance, the semantics for *car park* is car(x) $\wedge$ park(y) $\wedge$ UNSPEC(x,y) (simplifying slightly). UNSPEC(x, y) indicates an underspecified relationship between the two parts of the compound: this approach leads to over-generation even for compounds which are compositional (Copestake and Lascarides, 1997), but is worse for compounds where the elements are idiosyncratic, since the semantics is incorrect.[2]

It is clear that compound such as *car park* really should be recorded as idiosyncratic and we intend to do this in the MWE project. The practical problem with including established compound nouns in a computational lexicon is that it is a source of ambiguity because the productive noun-noun compound rule also applies. The LinGO ERG currently treats almost every compound noun as compositional (note

[2] This is on the assumption that the relation **park_rel** denotes 'normal' parks — the alternative is to treat it as denoting any use of *park* including car parks. In this case, the semantics is correct but is highly underspecified in use outside the compound as well as within it, which also leads to problems.

LEXEME_ID	BASEVERB	PARTICLE	TYPE	SEMPRED
look_up_v1	look_v1	up_noncomp_p1	v_particle_np_le	lookup_rel
tidy_up_v1	tidy_v1	up_completive_p2	v_reg_particle_np_le	

Table 3: Revised verb-particle database entries

```
easter_sunday := n_holiday_le &
  [ STEM < "easter", "sunday" >,
    SYNSEM.LOCAL.KEYS.KEY.NAMED 'easter_sunday ].

menlo_park_n1 := n_proper_le &
  [ STEM < "menlo", "park" >,
    SYNSEM.LOCAL.KEYS.KEY.NAMED 'menlo_park ].
```

Figure 3: ERG entries for compound nouns in TDL

how different this is from its current treatment of verb-particle constructions). The need for a productive rule for noun-noun compounding is clear, both on practical and theoretical grounds — the issue is how to avoid ambiguity by getting both productive and non-productive readings.

There are a variety of possible approaches:

- Blocking — do not construct compounds compositionally when a lexicalized compound exists. However, a lexicalized compound doesn't necessarily totally block productive formation, besides which implementing blocking requires additions to the formalism. Although the theoretical position that the compound should be blocked under normal circumstances is attractive, practically we have to allow for the fact that not all systems can implement this approach.

- Probabilities — giving the productive form low probability allows it to be dispreferred rather than totally blocked. However, it has to be dispreferred with respect to the lexicalized form, not globally. Again, not all systems support probabilities.

- Packing — assume both lexicalized and non-lexicalized readings, but use packing to reduce the ambiguity. The problem is that unpacking is eventually required for most applications.

- Compounds could be treated as productive, as now, but with lexicalized compounds checked for after parsing. The idea is that semantic relations such as _park_rel are treated as highly underspecified but default to the 'normal' non-compound use in non-compound environments. This approach can also be used for patterns of productive compounds (Copestake and Lascarides, 1997). One disadvantage, however, is that two different readings are really being combined in the syntax and compositional semantics: this will tend to add noise to the stochastic component for compounds such as *car park*.

If we think of the database as a device with which to generate the lexicon, rather than as a direct implementation of the TFS lexicon, the resolution of this problem can be system-specific. We can choose to treat compound MWE entries as normal lexical entries, or use them for semantic post-processing. In any case, the approach can be changed without changing the lexical database.

One further issue is that there is no principled upper bound for the number of elements in a compound noun, and there are some examples of compounds with three or more nouns in dictionaries, such as *daylight saving time*. There is clearly little problem representing longer compounds in a single relation in a database if a fixed limit on lexicalized compounds is stipulated. General principles of item familiarity would suggest that compounds with two elements are more likely to become lexicalized than those with more elements, but it is an open question whether there is a principled cut off point for lexicalized compounds or whether adopting an upper limit of, say, four elements, is simply an engineering compromise. Representations of compounds with indefinitely large numbers of elements are possible within relational databases, but involve the complication of using 'join' relations. We discuss this issue more generally below in the context of idioms.

4. Idioms

We make a strong distinction between decomposable and non-decomposable idioms, with the latter essentially being treated as words with spaces. The current LinGO grammar encodes a few decomposable idioms using the COMPKEY selection mechanism and other similar devices, but treats them as fully compositional (examples are *take advantage of* and *take care*). This is clearly not satisfactory for the more metaphorical idioms (e.g., *spill the beans* does not involve literally spillage or beans) and will only work for idioms which have a fixed syntactic head. The new approach proposed for encoding decomposable idioms in the MWE project is to treat them as semantically compositional, composed of idiomatic words which are only licensed in the context of an idiomatic phrase, which is represented by a phrasal entry.

For instance, Figure 4 shows a composite entry for the idiom *spill . . . beans*.[3] There are two idiomatic word entries

[3] The canonical form of this idiom is *spill the beans*, but vari-

```
i_spill := /spill_v1 & idiomatic &
[ SYNSEM.LOCAL.KEYS.KEY i_spill_rel ].

i_beans := /(bean + plural) & idiomatic &
[ SYNSEM.LOCAL.KEYS.KEY i_bean_rel ].

spill_the_beans := phrase &
[ SYNSEM.LOCAL.CONT.LISZT < [ PRED i_spill_rel,
                              ARG2 #y ],
                            [ PRED i_bean_rel,
                              ARG0 #y ],
                              ... >].
```

Figure 4: An idiom entry expressed in TDL

which default inherit from the corresponding non-idiomatic forms. The entry for the idiom phrase specifies that the variable introduced by *beans* has to instantiate the second argument position of *spill*, which allows both for the canonical order and for topicalization and other variation motivated by corpus evidence. The approach requires an minor extension to the typed feature structure formalism, because the check for licensing has to treat the relations in the CONT as unordered.

The approach is a variant of the one developed in Riehemann, 2001) and will not be discussed in detail. The point here is to discuss the difficulty of coming up with a suitable database representation. Problems arise because the number of idiomatic elements is not theoretically bounded and the range of interrelationships between the parts is very large. For instance, idioms like *six of one and half a dozen of the other* (and its variants) don't fit straightforwardly into any simple schema. On the other hand, the computational lexicographer really needs aids for constructing idiom lexicons, since the approach requires that idiomatic entries be written for words as well as phrases and these must be kept synchronized. Ideally we would like an integrated set of interactions between the lexicographer and the database interface to construct both the phrasal and word-level components of the idiom.

One possible solution rests on the observation that the majority of idioms fall into a fairly limited set of classes, although there is an indefinitely long tail of 'other' idioms. For instance, we can define a class of idioms that consist of an idiomatic verbal predicate taking an idiomatic nominal in object position. We could then define the idiom in terms of slots for the verb and head noun (possibly with an optional slot for the determiner). However, preliminary investigation suggests that the number of idiom classes that would be required to do this would be large, most would only be instantiated by a small number of idioms, and there would always be a substantial residue of idioms we could not capture properly. These idioms would have to be stored in the database as unanalyzed TDL expressions, which would limit their reusability. Another possibility is to develop a full account within the relational database, although this would involve a complex database structure, which might also not be easy to reuse. This issue remains to be resolved, since only experimentation on a reasonably large body of idioms can determine which solution is preferable in engineering terms.

5. Simplex words revisited

In the light of the discussion above, we can revisit the database for simplex words, since the perspective of treating the database as a first-class representation device also has implications there. In some ways, the use of the administrative information, such as source and date of entry, mentioned in §1., already involves such an idea. For instance, this makes it possible to construct different combinations of lexical information to allow for different redistribution restrictions on information arising from different sources. The database is generally useful in situations where information is available from an external source that we would like to use in principle, but cannot currently exploit in the TFS grammar. Such information can be stored in the database, and possibly used in an external module. An obvious example is the use of probabilities of different senses: not all systems which can process the ERG can make use of stochastic information, and those that do currently use weights rather than genuine probabilities, but we can usefully do lexical extraction experiments now and record information in the database, even if it has no current effect on processing. Similar remarks apply to dialect and register information — often this is very evident to a lexicographer and could be quickly recorded, even if not immediately utilized in the TFS representation. At least such information allows a filtered database to be constructed (e.g., one without obscenities).

6. Conclusions

This paper has outlined an approach to representing MWEs in a form which can support a precise HPSG but which is also reasonably transparent and reusable. We have discussed various issues in using a relational database in order to store lexical entries. A guiding principle is that, where possible, MWEs should be related to simplex entries. The approach relies on being able to identify classes of MWE that behave relatively regularly — idioms remain a problem because they are so diverse. Ongoing work involves refining the formal representation of the MWE classes and deciding on database structures. Productivity

ants with different determiners are found in corpora.

and semi-productivity play a major role in influencing the approach.

7. Acknowledgements

This research was supported in part by the Research Collaboration between NTT Communication Science Research Laboratories, Nippon Telegraph and Telephone Corporation and CSLI, Stanford University.

8. References

Copestake, Ann, 2002. *Implementing typed feature structure grammars*. CSLI Publications, Stanford.

Copestake, Ann and Alex Lascarides, 1997. Integrating symbolic and statistical representations: The lexicon-pragmatics interface. In *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics (ACL-97)*. Madrid.

Fellbaum, Christiane, 1998. Towards a representation of idioms in WordNet. In *Proceedings of the workshop on the Use of WordNet in Natural Language Processing Systems (Coling-ACL 1998)*. Montreal.

Nunberg, Geoffrey, Ivan A. Sag, and Thomas Wasow, 1994. Idioms. *Language*, 70:491–538.

Riehemann, Susanne, 2001. *A Constructional Approach to Idioms and Word Formation*. Ph.D. thesis, Stanford.

Villavicencio, Aline and Ann Copestake, 2002. Phrasal verbs and the LinGO-ERG. LinGO Working Paper No. 2002-01.