Virtual Polysemy

Antonio Sanfilippo, Kerima Benkerimi & Dagmar Dwehus*
SHARP Laboratories of Europe
Oxford Science Park, Oxford OX4 4GA, UK
{antonio,kerima,dagmar}@sharp.co.uk

Abstract

We present an approach to lexical knowledge representation where different uses of the same word can be conflated into a single *meta-entry* which encodes regularities about sense/usage extensibility. This approach makes it possible to solve lexical ambiguities by using contextual information during language processing to ground underspecified word entries, and can be efficiently implemented within a typed feature structure formalism.

1 Introduction

One of the central aspects of lexical knowledge, perhaps the most significant in characterizing the creative aspect of language use, is our ability to generate appropriate uses of words in context. This ability is usually exercized by manipulating semantic and/or syntactic properties of words to achieve desirable collocational settings. Some illustrative examples are given in (1) where

- *move* can be interpreted as a psychological verb when used transitively with a sentient direct object,
- *enjoy* can take either a noun or verb phrase complement when used in the *experience* sense (Pustejovsky, 1991, 1993; Briscoe, Copestake & Boguraev, 1990),
- *accord* is synonymous with either *agree* or *give/grant* depending on its valency (Poznański & Sanfilippo, 1993), and
- the occurrence of a directional argument with *swim* triggers a shift in aspectual interpretation.

(1) a. Please move your car
Her sadness moves him
b. John enjoys the book
John enjoys reading the book
c. The two alibis do not accord
They accorded him a warm welcome
d. John swam for hours
John swam across the channel

Although the precise mechanisms which govern lexical knowledge are still largely unknown, there is strong evidence that word sense extensibility is not arbitrary (Atkins & Levin, 1991; Pustejovsky, 1991, 1994; Ostler & Atkins, 1991). For example, the amenability of a transitive verb such as *move* to yield either a movement or psychological interpretation can be generalized to most predicates of caused motion (e.g. *agitate, crash, cross, lift, strike, sweep, unwind*) with the causer corresponding to the stimulus argument and the theme to the experiencer. Similarly, the option of either a noun or verb phrase complement for *enjoy* can be extended to many other psychological verbs with experiencer subjects (e.g. *hate, like, prefer*), and verbs of undirected motion in English (e.g. *carry, drive, float, push, run, swim, walk*) can subcategorize for an expression of completed path so as to yield a telic/directed interpretation (Talmy, 1985; Sanfilippo *et al.*, 1992; Sanfilippo, 1994). Moreover, the metonymical and metaphoric processes which are responsible for sense/usage extensions appear to be subject to crosslinguistic variation. For example, the "meat vs. animal" alternation that is found in English — viz. *feed the lamb* vs. *eat lamb* — is absent in Eskimo (Nunberg & Zaenen, 1992) as well as in Dutch where nominal compounding is used instead — e.g. *lam* vs. *lamvlees* (Copestake & Sanfilippo, 1993).

Examples of this sort show that our ability to extend word use in context is often systematic or conventionalized. As Pustejovsky and Boguraev (1993) point out, traditional approaches to lexical representation assume that word use extensibility can be modeled by exhaustively describing the meaning of a word through closed enumeration of its senses: each sense corresponds to a predefined context. This practice has largely characterized the compilation of dictionary entries in the lexicographic tradition and has consequently influenced the shape of computational lexicons since the large scale construction of such lexicons has typically involved semiautomatic knowledge acquisition from machine readable dictionaries (Carroll & Grover, 1989).

Word sense enumeration provides highly specialized lexical entries, but

- it fails to make explicit regularities about word sense extensibility which are necessary in promoting compactedness in lexical description,
- it is at odds with our ability to create new word uses in novel contexts, and
- it generates massive lexical ambiguity.

The use of lexical rules to generate different uses of a word from a kernel entry (Copestake & Briscoe, 1991; Sanfilippo, 1994) provides a principled alternative to word sense enumeration and can be made to cater for novel uses of words. However, it is not clear whether this practice can address the question of lexical ambiguity successfully as there is no known general control

*This work was carried out as part of the MT project at SHARP Laboratories of Europe. We would like to thank all members of the NLP group, and in particular Ian Johnson and Pete Whitelock, for helpful comments and advice.

regime on lexical rules which would deterministically restrict polysemic expansion without preempting the generation of possible word uses.

The goal of this paper is to show how a more dynamic approach to lexical specification can be used to tackle the problem of lexical ambiguity and at the same time to model creative aspects of word usage. In particular, our objective is to present ways in which word sense enumeration can be eschewed by conflating different word senses into a single *meta-entry* which allows sense/usage expansion without reliance on coercive operations such as lexical rules. This approach is implemented within a typed feature structure formalism where word sense conflation can be expressed in terms of lexical type underspecification: a word entry is associated with a lexical type having subtype extensions which describe possible uses of the word. This approach makes it possible to solve lexical ambiguities by using syntactic and semantic contextual information during language processing to ground underspecified word entries.

2 Lexical Polymorphism and Type Resolution

Our points of departure are (i) the polymorphic approach to lexical specification of Pustejovsky (1991, 1993) and (ii) the Attribute Logic Engine (ALE) formalism developed by Carpenter (1992a, 1992b).

Following Pustejovsky, we adopt an integrated multilayered representation of word meaning which incorporates salient aspects of world knowledge and where different uses of the same word are conflated into a single *meta-entry*. For example, a verb entry is assigned a lexical type which provides a specification of both argument and event structure including thematic and collocational (e.g. *qualia*) properties of its participants and can be extended to achieve contextual congruity (see below). In contrast with Pustejovsky, however, we do not use coercion as a main generative device to enforce sense extensions. True coercion involves type shifting which is operationally equivalent to a lexical rule (Pustejovsky, 1993). Consequently, the generation of sense extensions by coercion is ultimately of little avail in reducing lexical ambiguity, as was noted earlier for lexical rules.

Rather than using coercion, we encode lexical polymorphism by type underspecification and generate sense extensions using contextual information to ground lexical items. We provide such a specification of lexical structure within Carpenter's ALE using a HPSG-like grammar formalism (Pollard & Sag, 1992). This grammar formalism integrates a neo-Davidsonian approach to verb semantics (Parsons, 1990) where thematic roles are defined as prototypical notions (Dowty, 1991), see Sanfilippo (1993). Lexical types are arranged into an inheritance hierarchy with polymorphic types as intermediate nodes; each type can be associated with constraints expressed in terms of attribute-value pairs. For example, the lexical type of **synsem** for an intransitive verb such as *swim* is defined so as to subsume the types **iv_undir_synsem** and **iv_obl_dir_synsem** which characterize the two uses of the verb exemplified in (1d). This is shown in the type lattice fragment in Fig 1 where

- upper-case characters are used for attributes and bold lower-case for types (many details are omitted for ease of exposition)
- **dyn_eve** is a sort for non-stative eventualities (i.e. it subsumes processes and telic events)
- **pred** is either a lexical or logical predicate (**lex_pred**, e.g. *swim*; **log_pred**, e.g. *and*)
- **loc_chng** is a thematic sort which characterizes participants undergoing change of location
- **dir_prep** is a sort for prepositions which express a directed path (e.g. *to, across*).

Because *swim* in the lexicon is assigned the underspecified type **iv_undir_or_iv_obl_dir_synsem**, it can potentially combine with a complement and the subject arguments, or the subject only. In the first case, the complement list would be non-empty with its head instantiating a **pp_synsem** (prepositional phrase). The value for the path SYN:LOC:COMPS would thus resolve to the type **pp_comp_list** which as shown in (1) is the singleton list containing a **pp_synsem**. This is simply because **e_or_pp_comp_list** is defined as having subtypes **e_list** – the empty list – and **pp_comp_list** as shown in (5).

In a typed feature structure formalism with generalized recursive type resolution (Pollard & Sag, 1992:ch. 1; Carpenter, 1992a:ch. 15), the grounding of **e_or_pp_comp_list** to **pp_comp_list** would suffice to solve **iv_undir_or_iv_obl_dir_synsem** to **iv_obl_dir_synsem**. Instantiation for the head of the **comp_list** during parsing would then be sufficient to determine which use of the verb is contextually appropriate. Elegant as it might seem, however, generalized recursive type resolution leads to computational inefficiency. Moreover, if we assume that lexical entries are sort-resolved during rule application, it is difficult, perhaps impossible, to avoid multiple solutions for an underspecified lexical item when its rule context cannot lead to deterministic disambiguation. This would be the case when parsing a verb such as *bring* with a noun phrase complement. As can be inferred with reference to the three uses of the verb exemplified in (2), three solutions are possible until either the subject or the next complement is parsed:

(2) a. Mary brought Fido
    b. Mary brought Fido to the party
    c. Mary brought Fido a cookie

We tried to achieve a more efficient and deterministic treatment by developing special-purpose facilities which make available a guided approach the sort resolution. The basic intuition underlying such an attempt is that for every class of lexical ambiguity there is a specific word substructure whose instantiation is essential for disambiguation. For example, valency ambiguities for verbs can be generally resolved with reference to their complementation structure, as noted above for the two uses of *swim* in (1d). Likewise, the ambiguity of nominals such as *lamb* which can be used as either simple nouns or noun phrases in English (e.g. *feed the lamb* vs. *eat lamb*) can be contextually resolved with reference to determiner selection.

We used procedural attachments to rules to support contextually guided resolution of polymorphic lexical types. The ALE environment provides rather convenient facilities to carry out this implementation in the

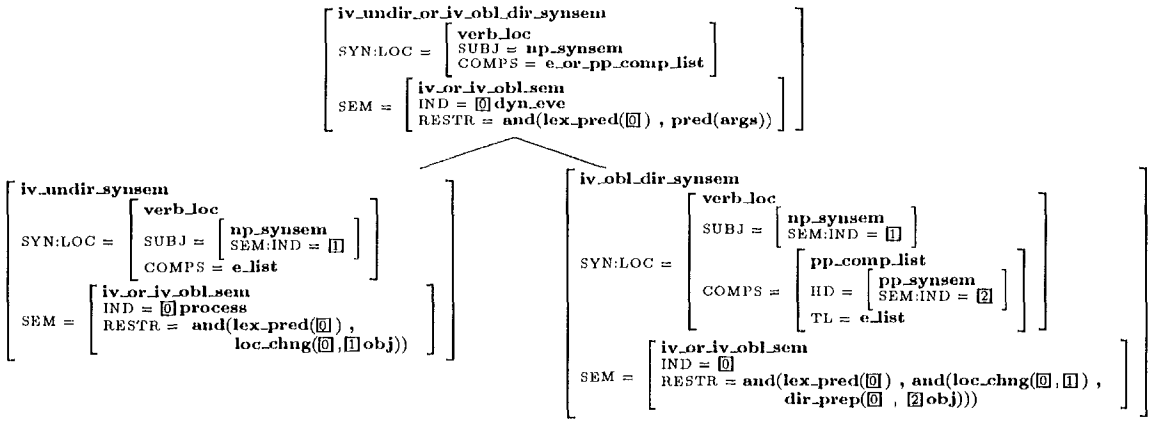


Figure 1: Using type subsumption to encode lexical polymorphism.

form of Prolog-style clauses where first-order terms are replaced with attribute-value descriptions. For example, given a definition of `list` as in (3a), the list-membership predicate can be defined as in (3b) where `X` is a typed feature structure (Carpenter, 1992b:ch. 4).

(3) a.
```
list sub [e_list, ne_list,
          comp_list, ...].
  e_list sub [].
  ne_list sub [ne_comp_list, ...]
              intro [hd:bot,
                     tl:list].
```
b.
```
member(X, hd:X) if true.
member(X, tl:Xs) if member(X,Xs).
```

Using the membership predicate above, we can define the ALE definite clause in (4) which would resolve polymorphic **verb_synsem** types by checking them against a list of unambiguous **synsem** types for consistency.

(4)
```
solve_head_type(Lex_Type) if
  member(Lex_Type, [iv_undir_synsem,
                    iv_obl_dir_synsem,...]).
```

`solve_head_type` can be integrated with grammar rules as shown schematically in Fig 2 so that a verbal head exhibiting valency ambiguity (e.g. **iv_undir_or_iv_obl_dir_synsem**) with contextual instantiation of its list of complements — **e_list** or **pp_comp_list**, as defined in (5) — would return a fully resolved FS (**iv_synsem** or **iv_obl_synsem** in Fig 1).

This way of carrying out lexical type resolution has computational overheads which tend to grow proportionally to the number of unambiguous lexical types. This is simply because lexical type resolution is done by unifying underspecified **synsem** FSs against a list of unambiguous lexical **synsem** FSs using the membership predicate: the longer the list, the heavier the computational overhead. With about thirty unambiguous verb types, we found that the disambiguation of polymorphic lexical types using `solve_head_type` with simple sentences was slower than enumeration of each distinct option through lexical disjunction — although the difference in performance tended to converge as we tried timing longer and more complex sentences.

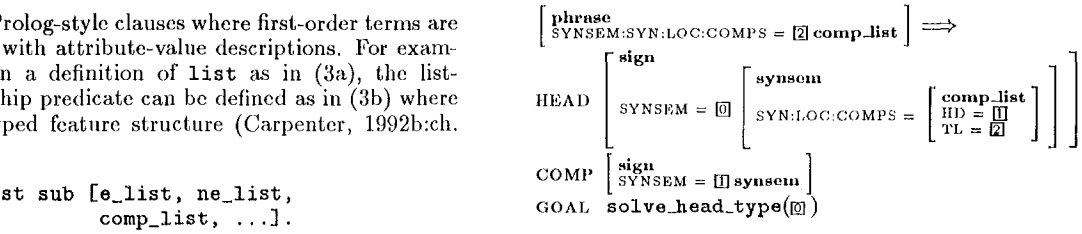


Figure 2: Simplified Head-Complement Rule with `solve_head_type` integrated as a goal.

(5)
```
comp_list sub [e_list, ne_comp_list,
               e_or_pp_comp_list].
  e_list sub [].
  ne_comp_list sub [pp_comp_list, ...]
               intro [hd:synsem,
                      tl:comp_list].
    e_or_pp_comp_list sub [e_list,
                           pp_comp_list].
      pp_comp_list sub []
                   intro [hd:pp_synsem,
                          tl:e_list].
```

Some improvements were obtained by eliminating the membership function and simply listing all possibilities as facts, e.g. `solve_head_type(iv_undir_synsem) if true.`, `solve_head_type(iv_obl_dir_synsem) if true.` However, we thought that better results yet could be achieved by exploiting conditions on constraint introduction rather than using unification with the list of unambiguous **synsem** FSs.

Since in ALE path values can be introduced as constraints, an attribute and its value can be used to retrieve the type at which that value was introduced:

```
| ?- restricts(Type,hd,pp_synsem).

Type = pp_comp_list
```

Our basic idea was to define a recursive definition of this facility and use it as a procedural attachment on rules to enhance lexical type resolution during language processing. For example, we could use the value for the head of the **comp_list** of a verb — as provided

in the course of rule application — and the path at which such value occurs to resolve the verb's lexical type, e.g.

```
| ?- rec_restricts(iv_or_iv_obl_synsem,
                   syn:loc:comps:hd:pp_synsem,
                   SubType).

SubType = iv_obl_synsem
```

This allowed us to carry out ambiguous lexical type resolution without having to check type compatibility against a list of unambiguous lexical types.

We devised a version of `rec_restricts` which given an ambiguous lexical type and the resolving contraint returns the appropriate grounded type by

1. retrieving all the minimal subtypes of the ambiguous type
2. collecting the constraints of each subtype into a list
3. returning the subtypes whose list of constraints include the resolving constraint.

The Prolog code for this algorithm is as shown below, where **sub**, **intro** and **cons** are ALE predicates which encode subsumption, feature introduction and constraint declaration.

```
rec_restricts(AmbigType,Cons,GroundedType):-
 findall(Type, (sub_type(AmbigType,Type),
         (Type sub [] ; Type sub [] intro _)),
         SubTypes),
  member(GroundedType,SubTypes),
  SubType cons Cons1,
  term_expansion(Cons1,[],[],ListCons),
  member_cons(Cons,ListCons).
```

`rec_restricts` is called from within `solve_head_type` which was redefined as a two place predicate whose arguments are: a (polymorphic) **synsem** type, and its resolving contraint as provided during the course of rule application, e.g.

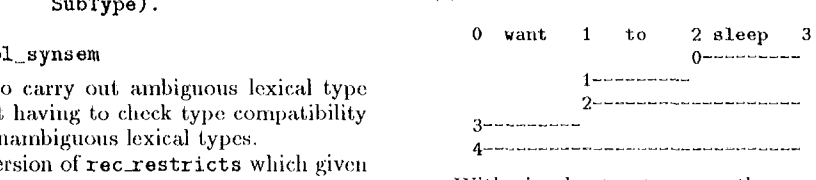
```
solve_head_type(iv_undir_or_iv_obl_dir_synsem,
                pp_synsem).
```

In the compiled code for `solve_head_type`, the unambiguous type given as output by `rec_restricts` (e.g. `iv_undir_synsem`) is used to resolve the input polymorphic type (`iv_undir_or_iv_obl_dir_synsem`) using unification of (atomic) **synsem** types rather than fully fledged FSs. This solution proved to be far more efficient than the previous one and never yielded worse results when compared to the enumeration of each distinct verb valency option through lexical disjunction.

3 Initial Results and Envisaged Improvements

Using the treatment outlined above, we have developed a type lattice covering all major complementation patterns for English and German (over 30 frames) with a variety of intermediate polymorphic types describing possible clusters of subcategorization options. At the same time, we have started to exploit the same technique for dealing with other cases of lexical ambiguity, such as the ability of nominals to function as either nouns or noun phrases, e.g. *John drank beer/a beer/beers/the beers.* Preliminary results are very encouraging. For example a verb such as *want* which can be used as either a transitive (*want a beer*), subject equi (*want to sleep*) or object raising verb (*want Mary to sleep*) will only produce a single chart edge when followed by a VP complement, e.g.

(6) `| ?- derivation([want,to,sleep]).`



With simple structures as the one in (6), the advantage in using polymorphic lexical types with sort resolution as compared to word sense enumeration by lexical disjunction is minimal even though fewer chart edges are built. This is because there is a constant overhead when doing polymorphic type resolution through `solve_head_type` which in these cases is equivalent to building a few more lexical edges. With more complex sentences, however, this overhead is soon offset, and the benefits of using lexical polymorphism become manifest. For example, the analysis of a sentence like *John likes that they want to come* using polymorphic verb types produced 23 edges and was about 15% faster than the analysis yielded using a lexicon with verb usage enumeration where 34 edges were built.

We are also confident that we can improve the performance of our approach in at least two regards.

First, we can reduce the computational effort currently used in ensuring that the input lexical type to `solve_head_type` has not been altered as a result of some previous rule application. Such a measure is needed, for example, when a verb with polymorphic type undergoes morphological combination before the head-complement rule applies. In this case, the semantics of the verb would be altered with a consequent loss of the original (polymorphic) lexical type. This would make lexical type resolution impossible. We must therefore avoid destructive modifications of the original lexical type while resolution of such type is still possible by introducing in the sign a structure where the semantics of the bound morpheme is stored until all verbal arguments are consumed. The stored semantics is then retrieved using procedural attachments. This retrieval is computationally expensive as it is carried out by means of procedural attachments, and we are now investigating the alternative of building the resulting semantics on line where it is currently stored.

Second, we can make lexical type resolution by `rec_restricts` more deterministic in those cases where the solving constraint does not lead to a unique solution, as discussed earlier with reference to the verb *bring*. In the lexicon, *bring* is assigned the polymorphic type **tv_or_tv_obl_or_ditrans_synsem** which subsumes the three uses of the verb exemplified in (2): **tv_synsem** in (2a), **tv_obl_ditrans_synsem** in (2b), and **ditrans_synsem** in (3b). Because the three subtypes are consistent with a direct object subcategorization, `rec_restricts` cannot provide a unique solution when parsing *bring* with a noun phrase complement. This is because `rec_restricts` carries out sort resolution of a polymorphic type by checking consistency of

the discriminating constraint against all minimal (most specific) subtypes of the polymorphic type. Consequently, `rec_restricts` would return three solutions for *bring* using the instantiation for the head of the **comp_list** to **np_synsem**, as would the use of generalized recursive constraint resolution. In our approach, however, this inadequacy can be easily redressed by

- changing `rec_restricts` so that sort resolution is done by returning the maximal (least specific) subtype of the input polymorphic type at which the discriminating constraint is introduced, and
- modifying the grammar so as to support such a change.[1]

As long as the same constraint is not introduced at several subtypes for each polymorphic type to be solved, these changes will ensure that sort resolution by `rec_restricts` is always deterministic.

4 Conclusion

If the computational analysis of natural language is to approach the ease with which language users manage the contextual determination of word usage, an approach to lexical ambiguity is needed which capitalizes on the regularity of sense extensions to avoid undiscriminated generation of word uses during sentence processing. Our proposal to achieve this objective is to use lexical polymorphism with deterministic contextual sort resolution within a type feature structure formalism. Such a proposal is based on the intuition that for each class of lexical ambiguity there is a word substructure whose incremental instantiation provides sufficient discriminating information to select a unique solution. We have shown how a first implementation of such an approach can be realized for the domain of verbal diatheses and envisaged how further refinements can be carried out to arrive at a full specification. Although it is too early to establish whether or not the approach can be made to handle all kinds of lexical ambiguity, initial results suggest that our treatment is effective, efficient and has natural applications in domains other than verbal diatheses.

References

Atkins B. & B. Levin (1991) Admitting Impediments. In U. Zernik (ed) *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, LEA Publishers, Hillsdale, New Jersey.

Briscoe T., A. Copestake & B. Boguraev (1990) Enjoy the Paper: Lexical Semantics via Lexicology. COLING-90.

Carpenter, B. (1992a) *The Logic of Typed Feature Structures.* Cambridge Tracts in Theoretical Computer Science 32, CUP, New York.

Carpenter, B. (1992a) *ALE: The Attribute Logic Engine User's Guide.* Ms, Carnagie Mellon University.

[1]This involves storing the putative grammatical link of a parsed argument (e.g. the direct object for *bring*) so that when a unique solution for the input polymorphic type is available this link can be used to establish the appropriate relation between complementation and argument structure — e.g. the direct object of *bring* would be interpreted as a goal in its ditransitive use (*bring Fido a cookie*) and as a theme otherwise (*bring Fido, bring Fido to the party*).

Carroll, J. & C. Grover (1989) The Derivation of a Large Computational Lexicon for English from LDOCE. In Boguraev, B. & Briscoe, T. (eds.) *Computational Lexicography for Natural Language Processing.* Longman, London.

Copestake, A. and A. Sanfilippo (1993) Multilingual Lexical Representation. In *Proceedings to the AAAI-93 Spring Symposium Series: Building Lexicons for Machine Translation*, Stanford, University, CA.

Copestake A. & T. Briscoe (1991) Lexical Operations in a Unification-Based Framework. In J. Pustejovsky & S. Bergler (eds) *Logical Semantics and Knowledge Representation*, Springer-Verlag.

Dowty, D. (1991) Thematic Proto-Roles and Argument Selection. Language 67, pp. 547-619.

Nunberg, G. & A. Zaenen (1992) Systematic Polysemy in Lexicology and Lexicography. EUROLEX-92.

Ostler N. and B. Atkins (1991) Predictable Meaning Shifts: Some Linguistic Properties of Lexical Implication Rules. In J. Pustejovsky & S. Bergler (eds) *Logical Semantics and Knowledge Representation*, Springer-Verlag.

Pollard, C. and I. A. Sag (1992) *Head-Driven Phrase Structure Grammar.* Ms. Stanford University, CA.

Poznański, V. and Sanfilippo A. (1993) Detecting Dependencies between Semantic Verb Subclasses and Subcategorization Frames in Text Corpora. In J. Pustejovsky and B. Boguraev (eds) *Acquisition of Lexical Knowledge from Text*, Proceedings of a SIGLEX Workshop, ACL-93, Columbus, Ohio.

Parsons, Terence (1990) *Events in the Semantics of English: a Study in Subatomic Semantics.* MIT press, Cambridge, Mass.

Procter, P. (1978) *Longman Dictionary of Contemporary English.* Longman, London.

Pustejovsky, J. (1991) The Generative Lexicon. *Computational Linguistics*, 17(4).

Pustejovsky, J. (1993) Linguistic Constraints on Type Coercion. Ms. Brandeis University.

Pustejovsky, J. & B. Boguraev (1993) Lexical Knowledge Representation and Natural Language Processing. *Artificial Intelligence* 63.

Sanfilippo, A. (1993) LKB Encoding of Lexical Knowledge. In Briscoe, T., A. Copestake and V. de Paiva (eds.) *Default Inheritance within Unification-Based Approaches to the Lexicon*, CUP.

Sanfilippo, A. (1994) Word Knowledge Acquisition, Lexicon Construction and Dictionary Compilation. COLING-94, Kyoto, Japan.

Sanfilippo, A., T. Briscoe, A. Copestake, M. Martí, M. Taulé and A. Alonge (1992) Translation Equivalence and Lexicalization in the ACQUILEX LKB. Proceedings of TM1-92, Montreal, Canada.

Talmy, L. (1985) Lexicalization Patterns: Semantic Structure in Lexical Form. In Shopen, T. (ed) *Language Typology and Syntactic Description 3. Grammatical Categories and the Lexicon*, CUP.