ACQUIRING LEXICAL DATA FROM MACHINE-READABLE DICTIONARY RESOURCES FOR MACHINE TRANSLATION

Mary S. Neff
Michael C. McCord

IBM T. J. Watson Research Center, P. O. Box 704, Yorktown Heights, New York 10598

ABSTRACT

This paper describes the direct and indirect automatic exploitation of multiple monolingual and bilingual machine-readable resources for acquiring lexical information, in (partial) real-time support of the English-to-German version of the Prolog-based machine translation system LMT. A detailed analysis of the structural properties of the lexical resources underlies our exploitation of them for the identification and extraction of syntactic and semantic characterizations of words, both as a subsystem of a machine translation project and as part of an overall enterprise of acquiring a computational lexicon.

INTRODUCTION

Building a lexicon for a large natural language processing system can be a labor-intensive enterprise. For this reason, many projects have turned to machine-readable published dictionaries. A project extensively exploiting a single machine-readable source for a natural language processing system is described by Boguraev and Briscoe (1987, 1989), features a restructured (LISPified) copy of the entire Longman Dictionary of Contemporary English (LDOCE) and which offers real time support for a general-purpose, wide coverage syntactic analyzer for English. In particular, the grammar codes of LDOCE are used for generating argument structures or frames. We extend that strategy along several dimensions. The English-to-German version of the Prolog-based machine translation system LMT (McCord 1989a,c,d), which now uses Slot Grammar (McCord 1980, 1989b, 1990) for source analysis, contains in its lexical access component a procedure for extracting slot frames for source and target languages primarily from a structured lexical data base (LDB) created from the Collins English-German (CEG) bilingual dictionary. Frames are derived from several different CEG fields, which exhibit greater diversity and considerably less formalization than the grammar codes in LDOCE. These are augmented, via mapping conventions, with English frames derived from the features in UDICT, a large encoded lexicon built for an English natural language processing application from multiple sources including LDOCE by batch jobs, analysis programs, and hand-edited lists (see Byrd, 1983; Klavans 1988a, I988b, 1989). During transfer, sense disambiguation and selection of target term is aided by access to a logical network of lexical relations currently represented by LDB's derived from monolingual English materials: LDOCE, Webster's Seventh Collegiate, and Collins Synonym Dictionary. Recognizing the incompleteness of even multiple machine-readable resources, we use dictionary analysis tools to add to a hand-built addendum lexicon.

THE LEXICAL ACCESS COMPONENT OF LMT

In its current (prototype) form, LMT allows the user to choose among different lexical configurations. For a given language pair, large stored lexicons may be organized in three separate parts, the source, transfer, and target lexicons; or all the information may reside in a single lexicon. Whatever the configuration, LMT does all lexical access at one time.

The lexical configuration described in this paper consists of (1) a largely hand-coded addendum lexicon, consulted first, containing source, transfer, and target information on words incorrectly or under generated from existing machine-readable resources - closed-class words and a core vocabulary; (2) a lexical data base (LDB) derived from the Collins English-German (CEG) bilingual dictionary, containing source and transfer information; (3) a large-coverage monolingual English dictionary (UDICT) encoded with feature information for use by natural language processing systems; and (4) a monolingual German lexicon derived from the Collins German-English (CGE) dictionary, with codings for noun and verb morphology. For simplicity, examples in this paper largely omit the morphology.

As all lexical access is done at once, the access component has procedures for coordinating and mapping the various sources and presenting the data in a standard format. The richness, diversity, and only semi-formalized nature of our machine-readable published resources have led us to build these procedures directly into LMT, so that we can experiment without regenerating the lexicon. System performance considerations will later argue in favor of collecting most of the resources and access mechanisms into a batch lexicon generator.

The Slot Grammar parser for LMT, as well as the transfer and generation components, require lexical information in an *internal form,* represented by Prolog clauses. For example, an internal-form lexical clause for a (possibly inflected or derived) source word `Word` is of the form

```
wframe(Word,Sense,Features,SlotFrame).
```

Internal-form clauses are created each time a sentence is processed — only for the words of that sentence — by the *lexical preprocessor,* which

handles morphology together with lookup in various user-selected external-form lexicons and LDB's. There is a *standard external form (SEF)* for LMT lexicons, made more abbreviated and compact than the internal form by a system of notational defaults. Addendum entries are stored in SEF; the conversions to internal form from the CEG LDB and UDICT also produce SEF as an intermediate step. SEF entries are of the form

```
Word < A1 < A2 < ... < An.
```

Nord is a source word in citation form. Each Ai of the entry is either a *source element* or a *transfer element.* An English-German example:

```
eat < v(obj)
    < t((animate&¬human).x ? fress I ess).
```

The first element **v(obj)** is a source element which says that *eat* has a direct object slot (subject slots are added by default in lexical preprocessing). The next element is a complex transfer element saying that *eat* transfers to *fressen* if the subject is animate and not human and otherwise into *essen.* It specifies a test on the arguments of the source word, where **(animate & ¬human)** tests the first argument (the subject) and x is an "I don't care" test on the second argument (the object). The question mark asks, "Is it true?" The first target word given, **fress**, is selected if the test succeeds; the *or else* **(|)** follows. The above example might be thought of as a conflation of two source element-transfer element pairs:

```
eat < v(obj) < t((animate&¬human).x ? fress)
    < v(obj) < t(ess).
```

The notion of conflating more complex pairs into a single entry will be taken up later.

Source elements specify mainly lexical categories and slot frames for the source word. The members of a slot frame are complement slots known to the source Slot Grammar and may be specified as obligatory slots or not. For example, slots specified for verbs include direct and indirect objects, finite and infinitive clause complements, and prepositional phrase complements. The grammar uses these in *slot filler rules, slot ordering rules,* and other types of rules (McCord 1989b,c,d, 1990). In addition, source elements can specify source morphological information, sense names, and semantic types. As indicated above, SEF notation has abbreviatory conventions with defaults. For example, if no sense name is specified (as in the 'eat' example above), then the sense name is taken to be the same as the citation form.

If one neglects source elements that specify irregular word forms, there is a one-to-one correspondence between source elements and transfer elements in a lexical entry. Specifically, for each source element **A** (having an associated lexical category and slot frame), there is exactly one transfer element **TA** which specifies the possible transfers of an occurrence of the source word with lexical analysis **A**. The transfer element **TA** can specify several target words, with tests to determine which one should be chosen.

The general form of a transfer element is

```
t(Tl | T2 | ... | Tn)
```

where each **Ti** is a *target word selector.*

The most general form of a target word selector is

```
Test ? TargetWord;TargetSlots.
```

The **Test** can consist of any Boolean combination of tests on the complements of the source word, or the word itself, or in fact any part of the source tree; and there is a convenient notation for expressing these tests. The test can be totally omitted from the notation, if no test is needed (for instance when there is only one target word selector). There is also a system of defaults for the **TargetSlots** specification, which often allows it to be omitted.

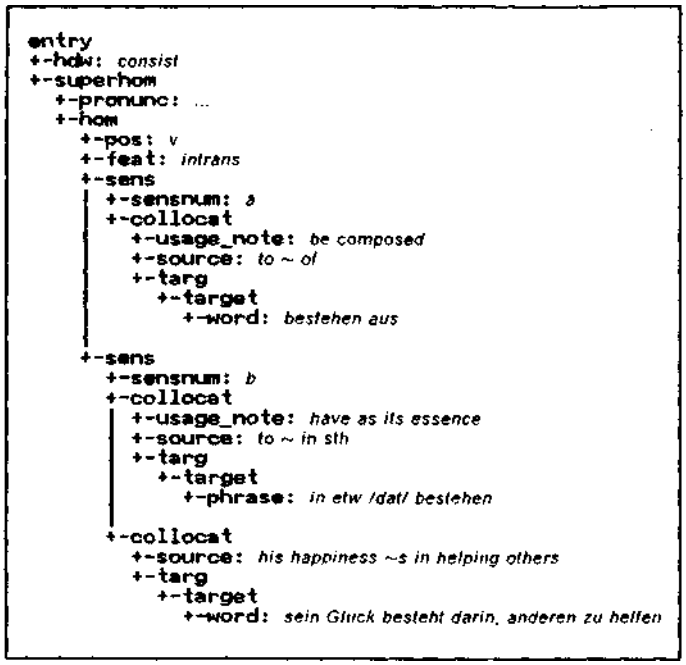
The sequence of target word selectors is treated like an if-then-else sequence; the first one whose test succeeds is applied. However, the tests normally can be specified in a declarative way so that order of target word selectors does not matter.

During lexical lookup, LMT gathers all available information about source and corresponding target terms (German noun class is derived from Collins German-English). For words not found in the addendum (not already in SEF format), other lexical resources are consulted; at the time of writing only CEG and UDICT are used in this step.

LEXICAL DATA BASE ACCESS

Conversion of the machine-readable type-setting tapes (or machine-readable dictionaries, MRD's) to LDB's was accomplished as a separate project (see Neff, 1989a, 1989b), with the goal of recovering implicit information and reconstructing elided fragments without any loss of data, so that the resulting LDB can be explored, queried, or referenced by an application. The sample CEG entry overleaf shows the characteristic hierarchical tree structure of entries.

The Lexical Query Language (LQL), an access method for LDB's (see Neff et. al., 1988, Byrd 1989), supports interactive querying by users and querying from a system. In our lexicon acquisition activities we use both modes, though the second is emphasized in this paper. Our lexical access component constructs LQL queries to get the contents of the several different fields in the CEG data base from which slot frames will be generated. For verbs, in addition to feature (e.g. transitivity), we focus on the **comp** (not shown in the sample entry) and colloc fields of the LDB, which contain prepositional phrase complements and collocations or example sentences respectively. Although in principle we would need to construct only one query for any given LDB to retrieve all the information of interest, we simplify here and focus only on collocations in verb entries. The query below says: for verb homographs of the word *consist,* extract



feature (transitivity), source collocation, and target collocation; format them as shown as Prolog clauses; and put the results on the console stack.

```
ENG-GERM:entry(. hdw: "consist";
                .superhom
                  .hom(
                    .tran: _trn;
                    .sens.collocat(
                      . source:_x;
                      .targ.target(
                        . word: _w; ) )
                    . pos:_cat;
                    . feat:_feat; ) )

CONDITION(_cat = "v")
FORMAT(e(ft("_feat").tran("_trn" ).src("_x" ).
         trg("_w")).
         %repeat all)
OUTPUT(CONSOLE STACK )
```

The result is:

```
e( ft("intrans").tran("").src("to ~ of").
     trg("bestehen aus")).
e( ft("intrans").tran("").src("to ~  in sth").
     trg("in etw /dat/ bestehen")).
```

How this query result will be used is the subject of the next section.

SLOT FRAME GENERATION

We generate SEF entries for all words not in the addendum. When we started, the frames were simple: source elements containing no arguments for nouns, only direct object arguments for verbs, and transfer elements for only the first translation in each syntactic category. As the frame generation component is being fleshed out, it returns a greater variety of source frames and tests. At the time of writing, we generate complex frames for verbs, having slots for NP objects and phrasal and clausal complements. To get these, we use the syntactic categories, prepositional phrase complements, collocations, and patterns of data in CEG and certain feature clusters in UDICT.

Collocations, or example sentences (CEG), are one source of frames. Complements (not illustrated) are another. Analysis of each source/target collocation pair returned by the query above yields two source/transfer pairs

```
consist < v(pl(of)) <  t(besteh ;  pl(aus))
        < v(pl(in)) <  t(besteh ;  pl(in)).
```

which are conflated to one SEF:

```
consist    < v(pl(of|in))
           <  t(prep(of)  ? besteh ;  pc(aus) |
              (prep(in)  ? besteh ;  pc( in )).
```

Because there are no translations for *consist* other than in the example sentences, we code the prepositional phrase complement as obligatory, or indicative of PP subcatcgorizalion (**pl**, rather than **p**). Other verbs with obligatory PP complements (labelled in Collins with the category *v prep),* are stored under a headword consisting of the verb plus preposition, so they must be searched for with another LDB access targeting a complex headword. Further evidence of PP subcategorization can be found in our monolingual materials, discussed below; for example, PP subcategorization is indicated on verbs marked in UDICT as **VPREP**. Nonobligatory PP complements are found in the several **comp** fields of the LDB. Clearly, then, different queries might target different data items to construct the same slot.

The collocations are also the source for slot frames describing more complex argument structures, such as phrasal and clausal complements. We use a small definite clause (DCG) grammar to parse CEG's English examples beginning with *to* ~. The grammar has no lexical lookup, but it recognizes

prepositions and slot fillers like *sb, sth.* If the English collocation parses successfully (i.e. it contains sufficiently general lexical items), we try to parse the corresponding German one with a similar DCG grammar with heuristics to recognize the verb and some adverbial complements. For example, the entry for *promise* appears in *CEG* (edited):

promise [...] 2 *vt (pledge)* versprechen; *(forecast, augur)* hindeuten auf *(acc).* **to ~ (sb) to do sth** (jdm) versprechen, etw zu tun; **to ~ sb sth, to ~ sth to sb** jdm etw versprechen; **~ me one thing** versprich mir eins; **to be ~d to sb** *(dated)* jdm versprochen sein *(old)* ; **I'm not promising anything but ...** ich will nichts versprechen, aber ...; ...
3 *vi* **(a)** versprechen. **(do you) ~?** versprichst du es?; **~!** *(will you ~)* versprich's mir, ehrlich?; *(I ~)* ehrlich!; **I'll try, but I'm not promising** ich werde es versuchen, aber ich kann nichts versprechen; **but you ~d!** aber du hast es doch versprochen!
(b) to ~ well vielversprechend sein; **this doesn't exactly ~ well** das ist nicht gerade vielversprechend.
4 *vr* **to ~ oneself sth** sich *(dat)* etw versprechen; **I've ~d myself never to do it again** ich habe mir geschworen, daß ich das nicht noch einmal mache.

From this, we generate the following pairs:

promise

(1) < v(obj) < t(versprech ; obj)
(2) < v(obj) < t(hindeut ; pl(auf,acc))
(3) < v(obj(inf),iobj)
< t(versprech ; obj(inf).iobj)
(4) < v(obj.iobj) < t(versprech ; obj.iobj)
(5) < v < t(versprech)
(6) < v(obj.iobj:reflex)
< t(versprech ; obj.iobj:reflex)

We conflate multiple frames (which incidentally could also be viewed as directed graphs) according to already well-known principles of graph unification (see, for example, Bouma, et al. 1988 and Uszkoreit 1986). If a graph (frame) A is more general in its information content than a graph (frame) B, then we say that A *subsumes* B. X *unifies* with Y, giving Z, if and only if Z is the most general graph (frame) that subsumes X and Y. Possible slots (e.g. **obj**) are of course more general than obligatory slots (e.g. **obj1**), and so subsume them. An object slot fillable by a finite clause or a noun phrase, coded as **obj(fin)**, subsumes one fillable only by a noun phrase, coded as **obj**. By applying the unification rules to all frames with the same target term, we conflate 1,3,4,5, and 6. If we only use the above data for the frames, (2) would be dropped since no tests distinguish it from (1). But if we use the dictionary's indicators (italicized information to label or discriminate among possibilities: here the synonyms *pledge* and *forecast),* we can code a distinction between the subsenses. In anticipation of LMT's one day being able to use these indicators for sense disambiguation, we might, for example, code a test for (2):

promise < v(obj)
< t(syn(forecast) . x . x ? hindeut ; pc(auf)).

Indicators also include domain labels, typical subjects, and typical objects; these may be coded, respectively, as a global test, (e.g. **engr sa**, meaning 'subject area engineering') or as local tests or the subject type field (e.g. **tsubj(river)**), or in the object slot (**obj(tobj(river))**). We describe later how LMT can use typical subjects and objects (the other indicators are not used yet).

Because CEG lacks some common English syntactic frames (the most frequent omissions involve parallel source and target constructions), we cannot use the bilingual dictionary as our only source of English syntactic frames. Because UDICT is our most broad-coverage English lexicon for natural language processing, we use UDICT features, singly or in constellation, to generate additional source frames. Using a set of heuristics which attempts to extract all the implied slot frames from UDICT's problematic flat feature representation (for UDICT, see Klavans and Wacholder, 1988), we get the following from UDICT:

promise < v(sub_control,*,obj(fin|inf).iobj).

If we now unify the English frame from UDICT with the results obtained from conflating 1,3,4,5,6 above, we get in SEF:

promise < v(sub_control,*,obj(fin|inf).iobj)
< t(versprech ; obj(fin|inf).iobj).

In this example, we have managed to get a significant feature (subject control) and create a new target slot (**obj(fin)**) from the application of the monolingual materials. Cases where a source slot is found in the bilingual dictionary but not in the monolingual one are gathered as a useful byproduct and routed back to the developers of the monolingual lexicon for possible improvements. Cases where unification is blocked because a slot or frame from the monolingual dictionary has no good match in the bilingual one are flagged for inspection and possible hand coding of bilingual entries. Because of the incompleteness of all our data, we remain suspicious of some of the results of the last unification step; errors or omissions discovered here are either used to improve the unification algorithm or hand-coded into the addendum.

Because the unification process is a general one, not necessarily tied to UDICT, we anticipate being able to factor in other English monolingual materials from which slot frames may be generated; at the same time we notice a possible penalty to pay for consultation of multiple sources: conflicting information. For example, there is a wealth of information on PP complements in the LDOCE grammar codes, which we will use in a planned link to the LDOCE LDB, but it does not always agree with what we find in CEG. The verb *agree,* according to CEG, subcategorizes with *to* and *on,* but the two senses of *agree with* are given in collocations as *~ with sth* and *sth ~s with sb,* and we cannot yet distinguish them automatically. LDOCE, however, indicates subcategorization with *with,* leaving *to, on, about,* and *with,* as other prepositions. For now, we indicate subcategorization

where one source has it, but not when a single dictionary is ambiguous.

Unused CEG collocations, such as *to promise well* and *to be promised to sb* do not produce frames at the moment but indicate directions of extension. For example, the explicit appearance of a passive construction in a collocation in the dictionary seems to indicate a special sense in English that might trigger something other than the passive in German. Other collocations pointing in the same direction are *to be located at, to be asphyxiated, to be called.* The following exploratory LQL query was submitted interactively, targeting English passives in CEG examples.

```
ENG-GERM:entry(.hdw: _h)
                    .superhom
                     .hom(
                        .sens.collocat (
                           .source: _x ;
                           .targ.target(
                              .phrase: _p;
                              .word: _w; ))
                        .pos: _syncat; ))
CONDITION(_syncat = "v";
          left<_x,8) = "to be ~d";
       I  left(_x,9) = "to be ~ed")
FORMAT(_h                    _x          _p _w;
       %repeat all)
OUTPUT(BE DASH_ED A)
```

An analysis of the query results largely supports this hunch; however they are rather complex: 126 of the German equivalents contain a construction with *sein* and either a past participle or an adjective, most often also with a prepositional phrase complement (e.g. *to be authorized : berechtigt sein);* 95 contain active verbs *(to be derailed : entgleisen);* 26 have a reflexive verb *(to be vexed about sth : sich über etw /acc/ ärgern);* 33 have true passives *(to be destroyed by fire : durch Brand vernichtet werden).*

Further analysis of these results and other results obtained from different kinds of interactive queries can lead to the definition of a complex problem like this, as well as pointing toward solutions. For the near term, however, these results suggest the construction of some (partially) hand-coded addendum entries.

In this paper we have limited our discussion to verbs; however generating slot frames for nouns, adjectives, and adverbs presents no additional problems of principle.

SENSE DISAMBIGUATION AND TARGET TERM SELECTION: USING A NETWORK OF LEXICAL KNOWLEDGE

Currently, sense disambiguation and correct target term selection use a common process that is part of lexical transfer. Tests of various kinds determine the selection of the target term and the shape of its frame. Consider our earlier example for *eat:*

```
eat < v(obj)
      < t( (animate&¬human).x ? fress  I  ess).
```

In this case the test is performed on the semantic type of the (underlying) subject of the verb. The absence of detailed semantic type information in our resources *(human* is coded but *animate* is not), together with the absence of a good theory of semantic types, prompts us to try to use the information that *is* available.

A human user of a bilingual dictionary is often guided in the selection of target term by the typical subject and typical object indicators. When the subject or object of the source sentence is not identical to the one given in the indicator, the human user applies knowledge of synonymy and taxonomy to make the right choice. In this section, we describe a strategy for automatically using the typical subject and typical object indicators, together with a network of lexical knowledge, represented by the Webster7 LDB, Collins Synonym LDB, LDOCE LDB, and a taxonym dictionary derived from Websters Seventh Collegiate (Chodorow and Klavans, 1990) for selecting among target terms.

We code the typical subject or object as **tsubj(Word)** or **tobj(Word)** and implement a test to find this typical subject among the following: the subject, a synonym, hypernym, hyponym or member of some nym-chain (of experimentally determined length and type) of the subject (cf. Binot 1987, Ravin 1990). The nym-chain could also be pursued the other way: beginning with typical subjects and trying to find the actual subject in one of their nym-chains. The network of nym relations is stored in several LDB's and is consulted during testing on subject or object.

Take for example the sentence, "The basement flooded yesterday." Our lexical entry, derived from CEG (ignoring the transitive senses and greatly simplifying the specification of multi-word translations) is:[1]

```
flood < v
      < t(tsubj(river).x  ?  'über die Ufer' :tret  |
          tsubj(bath).x  ?  überfließ |
          tsubj(cellar).x ? 'unter Wasser':steh |
          tsubj(garden | land).x ? überschwemmt:werd).
```

We wish to establish a unifying link between *basement* and one of the typical subjects in the entry. Hypernyms (from Webster's Seventh) *of basement* are *facade, interior, part.* Hyponyms are *semibasement* and *subbasement* - not very useful. There is no entry for *basement* in the Collins synonym dictionary. Synonym relations in standard dictionaries yield more useful information: Longman gives *cellar* as a synonym for *basement,* and Webster has *basement* as a synonym for *cellar.* The method would yield a translation of "Der Keller stand gestern unter Wasser." We may object to the difference in lexical aspect between

[1] Note that the *tests* appear in the target frames but refer to lexical items in the *source* language. Indeed, since source language sense disambiguation is postponed until transfer time, elements that might appear logically to be part of the source frame have migrated to the target in the form of tests.

flood and *unier Wasser stehen* in source sentence and translation — an indication of the limitations of our machine-readable resources.

The strategy for exploiting the typical object indicators is similar. Work elsewhere (see Ravin 1990, Braden-Harder, et al. 1990) indicates a maximum useful length of two for a nym-chain.

PROSPECTS

Other versions of LMT exist for other language pairs; the methodology described here will be applied to building lexicons for them as well.

ACKNOWLEDGEMENTS

We wish to acknowledge the significant contributions of Susanne Wolff to LMT's German noun morphology (based on Collins GE) and participation in the earliest link of LMT to CEG.

REFERENCES

Braden-Harder, L., and W. Zadrozny. 1990. "Lexicons for Broad Coverage Semantics", to appear in *Lexical Acquisition: Using On-Line Resources to Build a Lexicon,* edited by Zernik, Lawrence Erlbaum Associates (New Jersey).

Boguraev, B. K., R. J. Byrd, J. L. Klavans, and M. S. Neff. 1990. "From Structural Analysis of Lexical Resources to Semantics in a Lexical Knowledge Base," IBM Research Report RC 15427.

Boguraev, B. K. and E. J. Briscoe, Eds. 1989. *Computational Lexicography for Natural Language Processing.* Longman, Harlow and London (co-published in the United Stales with John Wiley and Sons, Inc., New York).

Boguraev, B.K. and E.J. Briscoe. 1987. "Large Lexicons for Natural Language Processing: Exploring the Grammar Coding System of LDOCE," *Computational Linguistics,* 13(3-4): 203-218.

Byrd, R. J. 1989. "LQL User Notes, An Informal Guide to the Lexical Query Language," IBM Research Report RC14853, IBM T.J. Watson Research Center, Yorktown Heights, N.Y.

Byrd, R. J. 1983. "Word Formation in Natural Language Processing Systems," *Proceedings of IJCAI-VIII*:704-706.

Binot J-L. and K. Jensen. 1987. "A Semantic Expert Using an Online Standard Dictionary," *Proceedings of IJCAI-87,* Milan, Italy.

Bouma, G., E. König, and H. Uszkoreit. 1988. "A flexible graph-unification formalism and its application to natural language processing," *IBM Journal of Research and Development,* Vol. 32, No. 2, 170-184.

Chodorow, M. S. and J. L. Klavans. 1990. "Locating Syntactic Patterns in Text Corpora," unpublished ms. IBM Research.

Klavans, J. I., and N. Wacholder. 1988. "Documentation of Features and Attributes in UDICT," IBM Internal Report RC14251. IBM T.J. Watson Research Center, Yorktown Heights, N.Y.

Klavans, J. L. 1988a. "COMPLEX: A Computational Lexicon for Natural Language Systems," *Proceedings of the 12th International Conference on Computational Linguistics.* Budapest, Hungary. (Also available as IBM RC13687)

Klavans, J. I . 1988b, to appear. "Building a Computational Lexicon using Machine Readable Dictionaries," to be published in *BUDALEX, Proceedings of the Third Congress of the. European Association for Lexicography.* Budapest, Hungary. (Also available as an IBM Internal Report RC14501)

McCord, M. C. 1980. "Slot Grammars," *Computational Linguistics,* vol. 6, pp. 31-43.

McCord, M. C. 1989a. "Design of LMT: A Prolog-based Machine Translation System," *Computational Linguistics,* vol. 15, pp. 33-52.

McCord, M. C. 1989b. "A New Version of Slot Grammar," Research Report RC 14506, IBM Research Division, Yorktown Heights, NY 10598.

McCord, M. C. 1989c. "A New Version of the Machine Translation System LMT," *J. Literary and Linguistic Computing,* vol. 4, pp. 218-229.

McCord, M. C. 1989d. "LMT," *Proceedings of MT Summit II,* pp. 94-99, Deutsche Gesellschaft für Dokumentation, Frankfurt.

McCord, M. C. 1990. "SLOT GRAMMAR: A System for Simpler Construction of Practical Natural Language Grammars," to appear in R. Studer (Ed.), *International Symposium on Natural Language and Logic,* Lecture Notes in Computer Science, Springer Verlag.

McCord, M. C. and S. Wolff. 1987. "The Lexicon and Morphology for LMT, A Prolog-Based MT System," IBM Internal Report RC59931).

Neff, M. S., R. J. Byrd, and O. A. Rizk. 1988. "Creating and Querying Hierarchical Lexical Data Bases," *Proceedings of the Second ACL Conference on Applied NLP,* Austin, Texas, 84-92.

Neff, M. S. and B. K. Boguraev. 1989a. "Dictionaries, Dictionary Grammars and Dictionary Entry Parsing," *Proceedings of the 27th Annual Meeting of the ACL,* Vancouver, B.C, June, 1989, 91-101.

Neff, M. S. and B. K. Boguraev. 1989b. "Dictionary Entry Parser — A Reference Guide," IBM Internal Report, IBM T.J. Watson Research Center, Yorktown Heights, N.Y.

Ravin, Yael. 1990. "Heuristics for Disambiguating and Interpreting Verb Definitions," to be published in *Proceedings of the 28th Meeting of the Association for Computational Linguistics ;* IBM Research Report RC I5655.

Uszkoreit, Hans. 1986. "Categorial Unification Grammars," CSLI Report No. CSLI-86-66.