BUILDING A RICH LARGE-SCALE LEXICAL BASE FOR GENERATION

Hongyan Jing and Kathleen R. McKeown and Rebecca Passonneau
Department of Computer Science
450 Computer Science Building
Columbia University
New York, N.Y. 10027
{hjing, kathy, becky} @cs.columbia.edu

May, 1997

Summary

Most large lexical resources have been developed with language interpretation in mind and can not be used directly for generation. We present a rich large-scale lexical base for generation, constructed by merging various linguistic resources. Our approach meets the needs of language generation systems by providing the facilities for mapping from semantic concepts to verb/sense pairs, for identifying the valid subcategorization forms for a given verb sense, and for representing alternations for paraphrasing power. Information from different resources enriches and constrains each other, so the final result is complete as well as accurate. We show by example how this lexical base can be intergrated into a generation package and how it simplifies development process while improving system performance.

TR Number: CUCS-016-97

Construction of large lexical resources for natural language processing has its roots in the use of lexicons for interpretation (Miller et al., 1990; Grishman et al., 1994). In contrast, most language generation systems still require hand encoding of lexical entries, restricting both coverage, portability, and paraphrasing power. The few exceptions use statistical analysis to construct lists of phrasal entries (Smadja and McKeown, 1991) or to build an n-gram model to allow probabilistic lexical choice (Knight and Hatzivassiloglou, 1995), but neither of these approaches aids in constructing the full lexical entries required for symbolic approaches to language generation.

Direct application of existing large-scale lexical resources to generation is not possible. Language generation requires choice of a word given a semantic concept as input, and the ability to consider both syntactic and semantic constraints for lexical choice. In this paper, we describe our work in building a large-scale lexical base for generation by automatically merging existing linguistic resources to produce the links between syntactic and semantic knowledge required in generation. We focus on verbs, since they play a more important role in deciding phrase structure and also have a more regular semantic structure. The database we construct is able to provide:

- A large variety of paraphrases for lexical items, including both common and rare transitivity alternations (Levin, 1993).
- Syntactic subcategorizations for verb senses, rather than for verbs.
- Mapping from semantic concepts to verb senses.
- Relations between semantic concepts, including hyponymy, antonymy, and entailment.

Merging resources is not a new idea and previous work has investigated integration of resources for machine translation and interpretation (Knight and Luk, 1994), (Rohini and Burhans, 1994), (Klavans et al., 1991). Our work differs in that we focus on resources for generation. Both the resources selected and the methodology used are quite different from others.

We show how this lexical base can be used in a generation system to significantly simplify development as well as improve system performance and reliability. In the following sections, we first introduce the lexical resources that we use. We then describe the algorithms for merging information from these resources and give some example applications of the lexical base.

1. RESOURCES AND PACKAGES

A generation lexicon must be indexed semantically in order to map from a word sense to a specific verb. Subcategorization patterns are typically linked to verbs, but often do not apply to all of the senses of a verb. Elsewhere (Jing et al., 1997), we present quantitative results on the degree to which different senses of a verb have distinct subcategorizations. Here we demonstrate how we automatically merge existing lexical resources to create a rich sense-based lexicon.

The following resources are used to build the lexical base: (1) English Verb Classes and Alternations (Levin, 1993) (EVCA), (2) COMLEX Syntax Dictionary (Grishman et al., 1994; Macleod and Grishman, 1995), (3) WordNet (Miller et al., 1990), and (4) the Brown Corpus (Kučera and Francis, 1967; Francis and Kučera, 1982). EVCA is of particular utility for generating paraphrases because it contains information about verbal diathesis, or transitivity alternations, as in *The girl sprayed water on the plants* → *The girl sprayed the plants with water*. However, EVCA has relatively few verbs (3,104), some of which are rare. COMLEX is a larger syntactic resource, with 5,583 verbs. It has the most complete and accurate representation of verb subcategorization of the resources used here, but contains little semantic information. WordNet is a word sense hierarchy, with each node consisting of a set of synonymous word senses (synsets). It has 11,364 verb synsets; arcs linking the nodes represent hyponymy, antonymy, and so on. The verbs in WordNet are classified into 15 semantic domains, such as change, cognition, and so on. WordNet also represents some syntactic information in sentence templates referred to as frames.

Our approach is to use overlapping information in the distinct resources as a basis for merging data. We manually develop a representation of the lexical syntactic knowledge in EVCA in a form that is compatible with COMLEX in order to facilitate automatic merging of the syntactic data. We do not use the semantic classification of verbs presented in EVCA because the semantic knowledge is largely implicit. WordNet frames are much less complete and accurate than the COMLEX subcategorizations, but provide a starting point for merging the rich semantic knowledge in WordNet with the syntactic data from EVCA and COMLEX. Thus an important issue we face is to ensure consistency across resources without loss of information.

2. ALGORITHMS FOR MERGING RESOURCES

System Architecture

Figure 1 shows the process used to merge the four lexical resources.

In step 1, we use our manually developed notation for encoding EVCA alternations to automatically derive properties for each verb. In step 2, we automatically merge EVCA verbs and their alternations with COMLEX verbs and their subcategorizations. In step 3, we manually build a look-up table to represent the compatibility of COMLEX/EVCA representations and WordNet frames, then use it to automatically merge WordNet synsets with COMLEX/EVCA verb entries. Finally, we use the Brown Corpus as a semantic concordance of WordNet and tag the result with frequency information. This eliminates some spurious entries resulting from merging WordNet and COMLEX/EVCA. Also, frequency information is useful during lexical choice for generation.

Information from the different resources enriches and constrains each other so the final result is complete as well as accurate. The merging process is automatic with just two exceptions: representing alternations and classes in the first step and building a compatibility matrix in the third step.

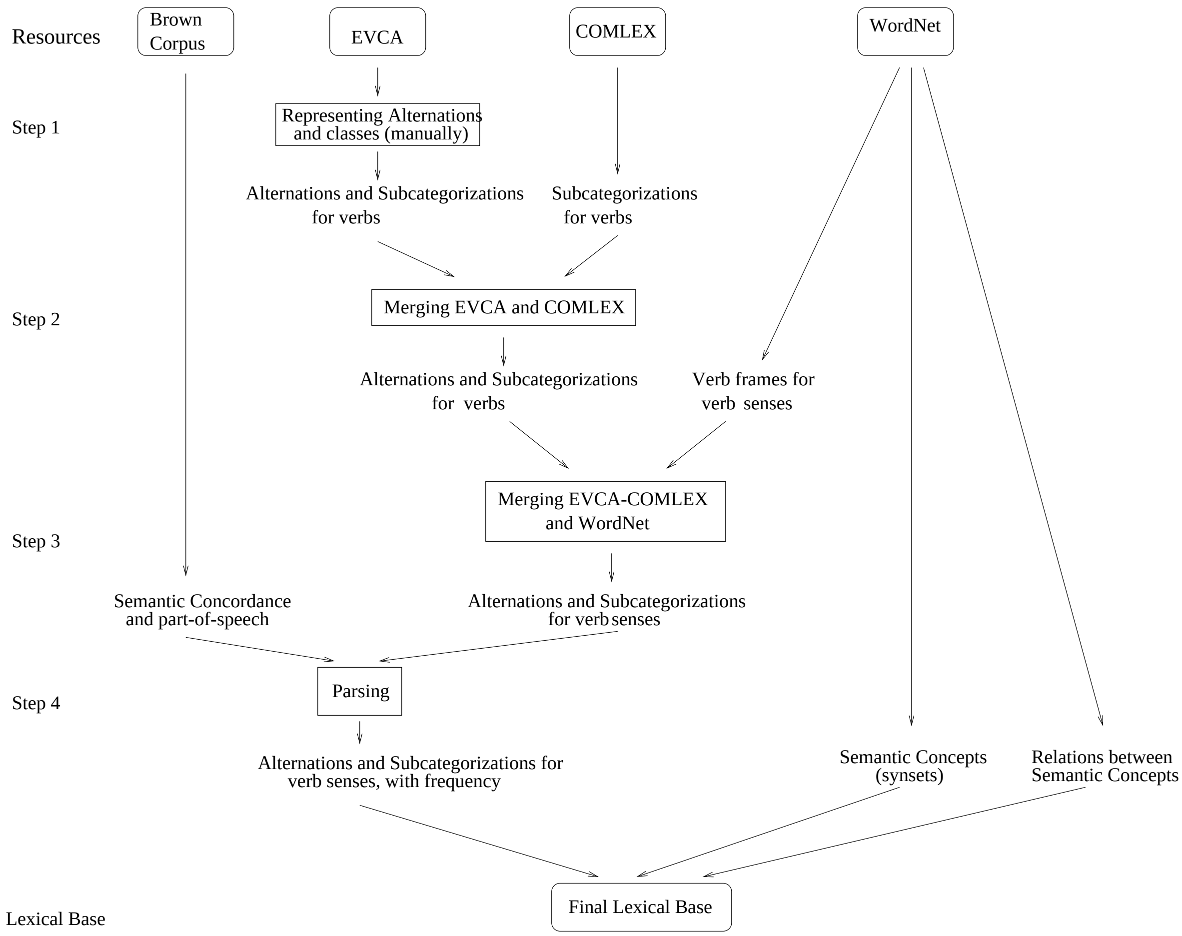

Figure 1: System Architecture

Alternations and Subcategorizations Acquisition from EVCA

EVCA has two major parts: the first contains the diathesis alternations and the second contains semantic verb classes based on shared alternations and subcategorizations. Diathesis alternations are illustrated in the book by example sentences and a list of verbs which have the same property. Verb classes are illustrated by member verbs followed by a list of alternations and subcategorizations for the class. Neither diathesis alternations nor verb classes are formatted or available on-line. The verb index, which is available online, contains each verb followed by the alternations and classes it belongs to. Our procedure is as follows:

1. Represent transitivity alternations.

A transitivity alternation may apply to only a few verbs (e.g, *Clear-Alternation*), or to hundreds of words (e.g., *There-Insertion*, as in *A girl appeared in the room* $\rightarrow$ *There appeared a girl in the room.*). A verb may have 0 or 1 alternations (e.g, the verbs *abandon* and *abate*), or up to more than 20 alternations (e.g, the verb *roll*).

We manually represent the syntactic pattern of each alternation using COMLEX subcategorization notation. 80 alternations were formatted in total.

The entry for the alternation *There-Insertion* is shown in Figure 2. The section number that the alternation is mentioned in appears first, then the name of the alternation, and finally, each category of the alternation, which corresponds to different alternating patterns. For example, *There-Insertion* alternations used by *Verb_of_Existence* as in category 1 and by *Run_Verbs* as in category 2 have different syntactic patterns. The keyword EXAMPLE represents the verb classes that fit in the category and SUBC represents the syntactic pattern of the alternation.

2. Represent verb classes.

Figure 3 shows our representation of the EVCA *Appear-Verb* class: all the member verbs (e.g, *appear*, *arise* etc) can occur intransitively, be followed by a locative PP, alternate by There-Insertion

```
(6.1 :ALT "There-Insertion"
     :1 alternating_verb
        (EXAMPLE ((be)
                  (verb_of_existence)
                  ...))
        (SUBC ((INTRANS THERE-V-SUBJ :ALT There-Insertion)
               (LOCPP THERE-V-SUBJ-LOCPP :ALT There-Insertion)))
     :2 manner_and_direction_specified
        (EXAMPLE ((run_verbs)
                  (roll_verbs)))
        (SUBC ((DIRPP THERE-V-DIRPP-SUBJ :ALT There-Insertion)))
     ...
     :6 change_of_state
        (EXAMPLE ((change_of_state_verbs)))
        (SUBC ((PP))))
```

Figure 2: Entry for the alternation *There-Insertion*.

etc. We represent all 191 verb classes.

3. Get alternations and subcategorizations for each verb.

Finally, we use the verb index to attach alternations and subcategorizations to specific verbs. In the verb index, each verb is followed by a list of the sections that it is mentioned in (e.g, *accept 2.2, 2.14, 13.5.2, 29.2.* Here 2.2 and 2.14 specify alternations; 13.5.2 and 29.2 specify verb classes.) We first retrieve the classes of a verb. From the class entry, we get the list of subcategorizations

```
(48.1.1 :CLASS "Appear_Verbs"
        :SUBC ((INTRANS)
               (LOCPP)
               (There-Insertion :CAT 1)
               (Locative_Inversion :CAT 1)
               (Causative/Inchoative :CAT 4)
               (Adj_Perfect_Part :CAT 1)))
```

Figure 3: Entry for the verb class *Appear_Verbs*.

```
(appear ((INTRANS)
         (LOCPP)
         (PP)
         (ADJ-PER-PART)
         (INTRANS THERE-V-SUBJ :ALT There-Insertion)
         (LOCPP THERE-V-SUBJ-LOCPP :ALT There-Insertion)
         (LOCPP LOCPP-V-SUBJ :ALT Locative_Inversion)))
```

Figure 4: Alternations and subcategorizations from EVCA for the verb *appear*.

```
(VERB    :ORTH "appear"
         :SUBC ((PP-TO-INF-RS :PVAL ("to"))
                (PP-PRED-RS :PVAL ("to" "of" "under" "against"
                                   "in favor of" "before" "at"))
                (EXTRAP-TO-NP-S)
                (INTRANS)
                (SEEM-S)
                (SEEM-TO-NP-S)
                (TO-INF-RS)
                (NP-PRED-RS)
                (ADJP-PRED-RS)
                (ADVP-PRED-RS)
                (AS-NP)))
```

Figure 5: COMLEX entry for the verb *appear*.

and the name of alternations, as in Figure 3. For each alternation, we retrieve the syntactic pattern from the alternation entry, as in Figure 2. Any alternations listed explicitly in the index that are not retrieved from the classes are added to the result.

Figure 4 shows the result from EVCA for the verb *appear*. It has 4 subcategorizations and 3 pairs of alternation patterns.

```
(appear ((PP-TO-INF-RS :PVAL ("to"))
        (PP-PRED-RS :PVAL ("to" "of" "under" "against" "in favor of"
                          "before" "at"))
        (EXTRAP-TO-NP-S)
        (INTRANS)
        (SEEM-S)
        (SEEM-TO-NP-S)
        (TO-INF-RS)
        (NP-PRED-RS)
        (ADJP-PRED-RS)
        (ADVP-PRED-RS)
        (AS-NP)
        (LOCPP)
        (INTRANS THERE-V-SUBJ :ALT There-Insertion)
        (LOCPP THERE-V-SUBJ-LOCPP :ALT There-Insertion)
        (LOCPP LOCPP-V-SUBJ :ALT Locative_Inversion)
        (ADJ-PER-PART)))
```

Figure 6: Alternations and Subcategorizations after merging information from EVCA and COMLEX for the verb *appear*.

```
appear   Sense 1   (give an impression)
                   * > Something ____s Adjective/Noun
                   * > Somebody ____s Adjective
                   * > Somebody ____s to INFINITIVE
         Sense 2   (become visible)
                   * > Something ____s
                   * > Somebody ____s
                   * > Something is ____ing PP
                   * > Somebody ____s PP
         ...
         Sense 8   (have an outward expression)
                   * > Something ____s Adjective/Noun
                   * > Somebody ____s Adjective
```

Figure 7: WordNet sense-syntax constraints for *appear*.

```
(appear ((1 ((PP-TO-INF-RS :PVAL ("to") :SO ((sb, –)) :FRE (1))
            (TO-INF-RS :SO ((sb, –)) :FRE (28))
            (NP-PRED-RS :SO ((sb, –)) :FRE (2))
            (ADJP-PRED-RS :SO ((sb, –) (sth, –)))) :FRE (11))
         (2 ((PP-TO-INF-RS :PVAL ("to") :SO ((sb, –) (sth, –)) :FRE (0))
            (PP-PRED-RS :PVAL ("to" "of" "under" "against" "in favor of"
                                  "before" "at")
                        :SO ((sb, –) (sth, –)) :FRE (0))
            (INTRANS :SO ((sb, –) (sth, –)) :FRE (30))
            (AS-NP :SO ((sb, –) (sth, –)) :FRE (2))
            (LOCPP :SO ((sb, –) (sth, –)) :FRE (12))
            (INTRANS THERE-V-SUBJ :ALT there-insertion
                                  :SO ((sb, –) (sth, –)) :FRE (30,0))
            (LOCPP LOCPP-V-SUBJ :ALT locative-inversion
                                :SO ((sb, –) (sth, –)))) :FRE (12,0))
         ...
         (8 ((NP-PRED-RS :SO ((sth, –)) :FRE (0))
            (ADJP-PRED-RS :SO ((sb, –) (sth, –)))))) :FRE (0))
```

Figure 8: Merging result for the verb *appear*.

Merging EVCA and COMLEX

COMLEX has a rather complete list of subcategorizations for each verb, but EVCA provides some subcategorizations omitted from COMLEX. Merging subcategorizations from EVCA and COMLEX also helps in checking the correctness of alternations.

Step one. For each verb, we maintain a subcategorization list and an alternation list. For a verb in both COMLEX and EVCA, we first copy all subcategorizations from COMLEX to the subcategorization list. Then for each subcategorization from EVCA, we compare it with the elements in the subcategorization list. If it is compatible with any element, it is ignored; otherwise it is added to the list. Because the subcategorizations in EVCA are associated with classes and alternations, they tend to be more general. For example, from EVCA, we get the subcategorization PP (i.e., the verb takes a prepositional subcategorization) for the verb *appear*. COMLEX has the subcategorization PP-PRED-RS (i.e, the verb takes a predicative prepositional subcategorization and the subject of the sentence is also the subject of the PP). Subcategorizations differing only in generality are considered compatible; here the more specific COMLEX form (e.g. PP-PRED-RS) is included in the final result.

For each alternation from EVCA, its alternating syntactic patterns are compared with elements in the subcategorization list. If both alternating patterns match some subcategorizations, the alternation is copied to the alternation list. Otherwise, an inconsistency between COMLEX and EVCA occurs and is written to the log file.

Step two. If a verb exists only in COMLEX or EVCA, it is copied to the result directly. COMLEX has a wider coverage than EVCA, with 5,583 and 3,104 verbs respectively. 337 verbs in EVCA are not present in COMLEX. After merging EVCA and COMLEX, we have 5,920 verbs in total.

Figure 5 shows the COMLEX entry for the verb *appear* and Figure 6 shows the result after merging EVCA and COMLEX. EVCA contributes two new subcategorizations (LOCPP and ADJ-PER-PART) to the final result. The overlapping subcategorization(INTRANS) only has one copy. The general form PP from EVCA is replaced by the more specific form PP-PRED-RS from COMLEX. The alternations are validated. In this case, there is no inconsistency.

Merging COMLEX/EVCA with WordNet

The result so far is still based on verbs, instead of verb senses. The verb *appear* has 8 senses in WordNet and a rich set of properties as shown in Figure 6. But for a specific sense, such as sense 4 (i.e, be apparent. e.g, *It appears that he is very gifted*), the sentence structure can only be *It appears (to somebody)+ that CLAUSE*.

To match properties for a verb to its senses, we make use of verb frame information in WordNet. The algorithm involves the following steps:

Step one. Manually construct a compatibility matrix for ECVA-COMLEX subcategorizations and frames from WordNet. We use 142 syntactic patterns, of which 92 are subcategorizations from COMLEX and the others are from EVCA. There are 35 verb frames in WordNet and each synset is marked with applicable frames. Due to the overly general specification of verb frames in WordNet, a subcategorization is considered compatible with a verb frame as long as it partially matches the frame. For example, the subcategorization PP is considered to be compatible with frames *Somebody _s PP*, *Something is _ing PP*, *Somebody _s on Something* etc. We have chosen to risk overgeneration rather than accidentally eliminating a valid property.

Step two. For each sense of a verb, we maintain an alternation and subcategorization list; a verb subcategorization is added to the verb sense subcategorization list if it is compatible with the

Wordnet frame for that sense along with the semantic type constraints on the subcategorization (e.g., *somebody* or *something*).

A verb alternation is considered applicable to a word sense only if all the alternating syntactic patterns (usually two) have matchable verb frames under that sense.

Step three. If a frame does not match any subcategorization for the verb, or a subcategorization or alternation for a verb does not match any frames of any senses, an inconsistency is noted in the log file. A syntactic pattern for a verb with no matching frame in WordNet is usually due to incompleteness of WordNet frames. A verb frame without a matching subcategorization is usually due to overgeneration WordNet frames.

Step four. Check the log file, adjust the compatibility matrix and go to step 2. This process is repeated over several passes. By regressively adjusting the compatibility matrix, we reduce the possibility of human judgment error and get more reliable results.

Figure 7 shows frames in WordNet for the verb *appear* and Figure 8 shows the result after assigning subcategorizations and alternations to each sense. Each sense of *appear* now has many fewer syntactic properties, as compared with the set for the entire verb *appear* (cf. Figure 6). In addition, the subcategorizations have been enriched with the general selectional type constraints (e.g., *somebody*, *something*) from the WordNet frames. Some of the syntactic properties assigned to a verb sense will now be spurious. For example, the second sense of the verb *appear*, to become visible, acquires PP-TO-INF-RS (e.g., *The busdriver appears to me to be falling asleep*; cf. Figure 8) We eliminate some of this overgeneration using frequency data.

Frequency Acquisition From Brown Corpus

We acquire frequency data for sense/subcategorization pairs from the Brown Corpus, which is tagged with part-of-speech and WordNet senses. If a sense/subcategorization pair does not occur, it is potentially spurious. We add frequency counts to the sense entries, which can then be used in constraining lexical choice (e.g., by preferring more frequent patterns). We use an incremental heuristic parsing strategy. First a sentence is divided into a sequence of components, roughly corresponding to distinct phrase types, based on the distribution of nouns and key parts of speech (e.g., PP, TO-INFINITIVE). Then we match the verb sense and the following or preceding constituent (e.g., depending on whether the verb is passive), to potential subcategorizations from our lexical base, and incrementally prune the subcategorizations by matching the next component until a single subcategorization is found. If none is found, we assume the subcategorizations of the verb sense are incomplete. After finding a match either in the subcategorizations for other senses of the verb, or for any verb, we add it to the subcategorizations for the current verb sense.

We checked a small set of verbs and it turns out that this simple algorithm works well.

3. APPLICATIONS

We have designed an architecture for integrating the lexical base into the FUF/SURGE package (Elhadad, 1991; Robin, 1994), a generation software environment developed at Columbia University, to save development time and improve system performance in generating paraphrases, and in syntactic and semantic error checking. We have tested each module of the architecture, demonstrating the conversion from an input semantic concept to a set of paraphrases, represented in the SURGE formalism. Input to SURGE, a wide-coverage systemic (Halliday, 1985; Winograd, 1983) grammar, is the thematic structure of the sentence to generate, represented using a systemic *process* hierarchy.

A FUF/SURGE lexical chooser maps a conceptual representation of what is to be said into the thematic structure that SURGE expects as input. During this task, it must map an input conceptual relation to a verb (e.g, *like*, *love* etc), specify the process type (e.g., *material*, *verbal*), and map conceptual entities in the relation to the process type participants. Participants are the systemic equivalent of thematic roles. For example, the participants of a material process are *agent*, *affected*, while participants of a verbal process are *sayer*, *addressee*, *verbalization*.

The hyponym relations in WordNet can help us decide the process type of a verb sense automatically. We speculate that: (1) Verb senses in a synset have the same process type; and (2) a synset inherits the process type from its superordinate synset in WordNet. Evidence supporting this second point comes from a check of all 720 verbs from 399 synsets in the competition semantic domain in WordNet; all were members of the *material* process type. Full implementation would require a mapping relation from each root node in WordNet corresponding to a verb synset to its FUF/SURGE process type (N=16).

Our lexical base can provide several outputs for the same input. For the input corresponding to *A girl appeared in the room*, two more paraphrases can be generated using the alternation data: *There appeared a girl in the room* by the *There-Insertion* alternation and *In the room appeared a girl* by *Locative-Inversion*.

Given our lexical base, a FUF/SURGE lexical chooser can be greatly simplified. After mapping an input conceptual relation to a WordNet synset, the corresponding SURGE process type and participants can be retrieved automatically. Our lexical base provides subcategorizations for the sense that can be used as a check on whether the input information is coherent. This can avoid overgeneration by ruling out invalid combinations of subcategorizations with specific senses. Paraphrasing power can be enhanced by automatically generating SURGE input for each alternation. Which paraphrase is selected can be based on the frequency information for the domain or we could provide a browser to allow the system developer to interactively select an appropriate paraphrase.

4. EVALUATION AND FUTURE WORK

Due to lack of another large corpus tagged with WordNet senses, large-scale evaluation of the information acquired from merging resources was not conducted at this stage. The sparse data problem is more significant in our evaluation because we need occurrence information for each subcategorization of each verb sense. The COMLEX group recently tagged instances of 750 verbs in a large corpus with their subcategorization class (Macleod et al., 1996; Macleod et al., 1995) and we plan to use that to evaluate a subset of verbs in the future.

The use of such a large-scale lexical base in generation makes lexical choice over a broad range of words possible, but also points out the need for a systematic approach to lexical choice constraints. Many distinct types of information play a role in lexical choice, such as syntax, frequency, argument structure, and collocational patterns. How to generalize the lexical choice process, e.g., by determining the ordering of different types of constraints, and how to represent the input to the lexical chooser or generator, are topics we plan to explore.

5. CONCLUSIONS

We have described the process of merging various linguistic resources to build a large-scale lexical base for generation. The resources used both enrich and constrain each other, so that the final lexical base is relatively complete and accurate. Our approach meets language generation needs by providing the facilities for mapping from semantic concepts to verb/sense pairs, for identifying the valid subcategorization forms for a given verb sense, and for representing alternations for paraphrasing power. To demonstrate its utility for generation, we have shown how the lexical base can be linked with the FUF/SURGE generation package.

REFERENCES

M. Elhadad. 1991. Fuf user manual - version 5.0. (Tech. Rep. CUCS-038-91) CUCS-038-91, Columbia University - Computer Science.

W. N Francis and H. Kučera. 1982. *Frequency Analysis of English Usage: Lexicon and Grammar.* Houghton Mifflin, Boston, MA.

Ralph Grishman, Catherine Macleod, and Adam Meyers. 1994. COMLEX syntax: Building a computational lexicon. In *Proceedings of COLING-94*, Kyoto, Japan.

M.A.K Halliday. 1985. *An Introduction to Functional Grammar.* Edward Arnold, London.

Hongyan Jing, Vasileios Hatzivassiloglou, Rebecca Passonneau, and Kathleen McKeown. 1997. Investigating complementary methods for verb sense pruning. In *Proceedings of ANLP97 Lexical Semantics Workshop*, Washington, D.C., April.

J. Klavans, R. Byrd, N. Wacholder, and M. Chodorow. 1991. Taxonomy and polysemy. Technical Report Research Report RC 16443, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, NY 10598.

Kevin Knight and Vasileios Hatzivassiloglou. 1995. Two-level, many-paths generation. In *Proceedings of the 33st Annual Meeting of the Association for Computational Linguistics*, Boston, MA, June.

Kevin Knight and Steve K. Luk. 1994. Building a large-scale knowledge base for machine translation. In *In AAAI-94 proceedings.*

H Kučera and W. N. Francis. 1967. *Computational Analysis of Present-day American English.* Brown University Press, Providence, RI.

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation.* University of Chicago Press, Chicago, Illinois.

Catherine Macleod and Ralph Grishman, 1995. *COMLEX Syntax Reference Manual.* Proteus Project, New York University.

Catherine Macleod, Adam Meyers, and Ralph Grishman. 1995. Tagging as a means of refining and extending syntactic classes. In *Proceedings of the 1995 AAAI Spring Symposium on the Lexicon*, Stanford University.

Catherine Macleod, Adam Meyers, and Ralph Grishman. 1996. The influence of tagging on the classification of lexical complements. In *Proceedings of COLING-96*, Copenhagen, Denmark.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography (special issue)*, **3**(4):235–312.

Jacques Robin. 1994. *Revision-Based Generation of Natural Language Summaries Providing Historical Background: Corpus-Based Analysis, Design, Implementation, and Evaluation.* Ph.D. thesis, Department of Computer Science, Columbia University, New York. Also Technical Report CU-CS-034-94.

R. K. Rohini and D. T. Burhans. 1994. Visual semantics: Extracting visual information from text accompanying pictures. In *In AAAI-94 proceedings*.

Frank Smadja and Kathleen R. McKeown. 1991. Using collocations for language generation. *Computational Intelligence*, 7(4), December.

T. Winograd. 1983. *Language as a Cognitive Process*. Addison-Wesley, Reading, Ma.