Generation from Lexical Conceptual Structures

David Traum and **Nizar Habash**
Institute for Advanced Computer Studies
University of Maryland College Park
{traum,habash}@cs.umd.edu

1 Introduction

This paper describes a system for generating natural language sentences from an interlingual representation, Lexical Conceptual Structure (LCS). This system has been developed as part of a Chinese-English Machine Translation system, however, it promises to be useful for many other MT language pairs. The generation system has also been used in Cross-Language information retrieval research (Levow et al., 2000).

One of the big challenges in Natural Language processing efforts is to be able to make use of existing resources, a big difficulty being the sometimes large differences in syntax, semantics, and ontologies of such resources. A case in point is the interlingua representations used for machine translation and cross-language processing. Such representations are becoming fairly popular, yet there are widely different views about what these languages should be composed of, varying from purely conceptual knowledge-representations, having little to do with the structure of language, to very syntactic representations, maintaining most of the idiosyncracies of the source languages. In our generation system we make use of resources associated with two different (kinds of) interlingua structures: *Lexical Conceptual Structure* (LCS), and the *Abstract Meaning Representations* used at USC/ISI (Langkilde and Knight, 1998a).

2 Lexical Conceptual Structure

Lexical Conceptual Structure is a compositional abstraction with language-independent properties that transcend structural idiosyncrasies (Jackendoff, 1983; Jackendoff, 1990; Jackendoff, 1996). This representation has been used as the interlingua of several projects such as UNITRAN (Dorr, 1993) and MILT (Dorr et al., 1997).

An LCS is a directed graph with a root. Each node is associated with certain information, including a *type*, a *primitive* and a *field*. The type of an LCS node is one of *Event, State, Path, Manner, Property* or *Thing*. There are two general classes of primitives: closed class or structural primitive (e.g., `CAUSE, GO, BE, TO`) and open class primitives or constants (e.g., `reduce+ed, textile+, slash+ingly`). suffixes such as `+, +ed, +ingly` are markers of the open class of primitives. primitives have a Examples of fwields include `Locational, Possessional, Identificational`.

An LCS captures the semantics of a lexical item through a combination of semantic structure (specified by the shape of the graph and its structural primitives and fields) and semantic content (specified through constants). The semantic structure of a verb is something the verb inherits from its Levin verb class whereas the content comes from the specific verb itself. So, all the verbs in the "Cut Verbs - Change of State" class have the same semantic structure but vary in their semantic content (for example, chip, cut, saw, scrape, slash and scratch).

The lexicon entry or Root LCS (RLCS) of one sense of the Chinese verb *xue1_jian3* is as follows:

(1)

```
(act_on loc (* thing 1) (* thing 2)
        ((* [on] 23) loc (*head*) (thing 24))
        (cut+ingly 26)
        (down+/m))
```

The top node in the RLCS has the structural primitive `ACT_ON` in the locational field. Its subject is a star-marked LCS (or an unspecified LCS) with the restriction that a filler LCS be of the type thing. The number "1" in that node specifies the thematic role: in this case, agent. The second child node is in an argument position and needs to be of type thing too. The number "2" stands for theme. The last two children specify the manner of the `locational act_on`, that is "cutting in a downward manner". The RLCS for nouns are generally much simpler since they include only one root node with a primitive. For instance (US+) or (quota+).

The meaning of complex phrases is captured through a CLCS – composed LCS. This is constructed "composed" from several RLCSes corresponding to individual words. In the composition process that starts with a parsed tree of the input sentence, all the obligatory positions in a RLCS are filled with other RLCSes. For example, the three RLCSes we have seen already can compose to give the CLCS for the sentence: *United states cut down (the) quota*

(2)

```
(act_on lo c (us+) (quota+)
       ((* [on] 23) loc (*head*) (thing 24))
        (cut+ingly 26)
        (down+/m))
```

CLCS structures can be composed of different sorts of RLCS structures, corresponding to different words. A CLCS can also be decomposed on the generation side in different ways depending on the RLCSes of the lexical items in the target language. For example, the CLCS above will match a single verb and two arguments when generated in Chinese (regardless of the input language). But it will match four lexical items in English: cut, US, quota, and down, since the RLCS for the verb "cut" in the English lexicon does not include the modifier `down`:

(3)

```
(act_on loc (* thing 1) (* thing 2)
       ((* [on] 23) loc (*head*) (thing 24))
(cut+ingly 26))
```

The rest of the examples in this paper will refer to the slightly more complex CLCS of the sentence *The United States unilaterally reduced the China textile export quota* in (4) below, which roughly corresponds to "The United States caused the quota (modified by china, textile and export) to go identificationally (or transform) towards being at the state of being reduced." This LCS is presented without all the additional features for sake of clarity. Also, it is actually one of eight possible LCS compositions produced by the analysis component from the input Chinese sentence.

(4)

```
(cause (us+)
   (go ident (quota+ (china+)
                     (textile+)
                     (export+))
       (to ident (quota+ (china+)
                         (textile+)
                         (export+))
           (at ident (quota+ (china+)
                             (textile+)
                             (export+))
                (reduce+ed))))
  (with instr (*HEAD*) nil)
  (unilaterally+/m))
```

3 The Generation System

Since this generation system was developed in tandem with the most recent LCS composition system, and LCS-language and specific lexicon extensions, a premium was put on the ability for experimentation along a number of parameters and rapid adjustment on the basis of intermediate inputs and results to the generation system. This goal encouraged a modular design, and made lisp a convenient language for implementation. We were also able to successfully integrate components from the Nitrogen Generation System (Langkilde and Knight, 1998a; Langkilde and Knight, 1998b).

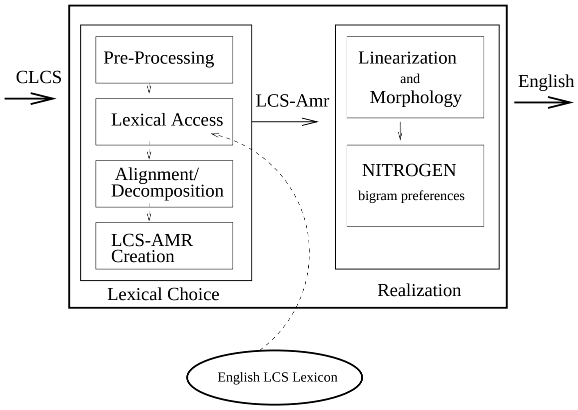

Figure 1: Generation System Architecture

The architecture of the generation system is shown in Figure 1, showing the main modules and sub-modules, and flow of information between them. The first main component translates, with the use of a language specific lexicon, from the LCS interlingua to a language-specific representation of the sentence in a modified form of the AMR-interlingua, using words and features specific to the target language, but also including syntactic and semantic information from the LCS representation. The second main component produces target language sentences from this intermediate representation. We will now describe each of these components in more detail.

The input to the generation component is a text-representation of a **CLCS**, the Lexical Conceptual Structure corresponding to a natural language sentence. The particular format, known as *long-hand* is equivalent to the form shown in (4), but making certain information more explicit and regular (at the price of increased verbosity). The Long-hand CLCS can either be a fully language-neutral interlingua representation, or one which still incorporates some aspects of the source-language interpretation process. This latter may include grammatical features on LCS nodes, but also nodes, known as *functional nodes*, which correspond to words in the source language but are not LCS-nodes themselves, serving merely as place-holders for feature information. Examples of these nodes include punctuation markers, coordinating conjunctions, grammatical aspect markers, and determiners. An additional extension of the LCS input language, beyond traditional LCS is the in-place representation of ambiguous sub-trees as a *possibles* node, which has the various possibilities represented as its own children.

Thus, for example, the following structure (with some aspects elided for brevity) represents a node that could be one of three possibilities. In the second one, the root of the sub-tree is a functional node, passing its features to its child, `COUNTRY+`:

(5)

```
(:POSSIBLES -2589104
    (MIDDLE+ (COUNTRY+ ( DEVELOPING+/P)))
    (FUNCTIONAL (POSTPOSITION AMONG)
                (COUNTRY+ (DEVELOPING+/P)))
    (CHINA+ (COUNTRY+ ( DEVELOPING+/P)))
)
```

3.1 Lexical Choice

The first major component, divided into four pipelined sub-modules, as shown in Figure 1 transforms a CLCS structure to what we call an LCS-AMR structure, using the syntax of the abstract meaning representation (AMR), used in the Nitrogen generation system, but with words already chosen (rather than more abstract Sensus ontology concepts), and also augmented with information from the LCS that is useful for target language realization.

3.1.1 Pre-Processing

The pre-processing phase converts the text input format into internal graph representations, for efficient access of components (with links for parents as well as children), also doing away with extraneous source-language features, converting, for example, (5) to remove the functional node and promote `COUNTRY+` to be one of the possible sub-trees. This involves a top-down traversal of the tree, including some complexities when functional nodes without children (which then assign features to their parents) are direct children of possibles nodes.

3.2 Lexical Access

The lexical access phase compares the internal CLCS form to the target language lexicon, *decorating* the CLCS tree with the RLCSes of target language words which are likely to match sub-structures of the CLCS. In an off-line processing phase, the target language lexicon is stored in a hash-table, with each entry keyed on a *designated primitive* which would be a most distinguishing node in the RLCS. On-line decoration then proceeds in two step process, for each node in the CLCS:

(6) a. look for RLCSes stored in the lexicon under the CLCS node's primitives

b. store retrieved RLCSes at the node in the CLCS that matches the root of this RLCS

Figure 2 shows some of the English entries matching the CLCS in (4). For most of these words, the designated primitive is the only node in the corresponding LCS for that entry. For *reduce*, however, `reduce+ed` is the designated primitive. While this will be retrieved at the `reduce+ed` node from (4), in step (6)a, in (6)b, the LCS for "reduce" will be stored at the root node of (4) (`cause`).

```
(:DEF_WORD "reduce"
 :CLASS "45.4.a"
 :THETA_ROLES ((1 "_ag_th,instr(with)"))
 :LCS (cause (* thing 1)
        (go ident (* thing 2)
            (toward ident (thing 2)
                    (at ident (thing 2)
                        (reduce+ed 9))))
        ((* with 19) instr (*head*)
     (thing 20)))
 :VAR_SPEC ((1 (animate +))))

(:DEF_WORD "US" :LCS (US+ 0))

(:DEF_WORD "China" :LCS (China+ 0))

(:DEF_WORD "quota" :LCS (quota+ 0))

(:DEF_WORD "WITH"
 :LCS (with instr (thing 2) (* thing 20)))

(:DEF_WORD "unilaterally"
 :LCS (unilaterally+/m 0))
```

Figure 2: Lexicon entries

The current English lexicon contains over 11000 RLCS entries such as those in Figure 2, including over 4000 verbs and 6200 unique primitive keys in the hash-table.

3.3 Alignment/Decomposition

The heart of the lexical access algorithm is the *decomposition* process. This algorithm attempts to align RLCSes selected by the lexical access portion with parts of the CLCS, to find a complete covering of the CLCS graph. The main algorithm is very similar to that described in (Dorr, 1992 1993), however with some extensions to be able to also deal with the in-place ambiguity represented by the *possibles* nodes.

The algorithm recursively checks a CLCS node against corresponding RLCS nodes coming from the lexical entries retrieved and stored in the previous phase. If significant incompatibilities are found, the lexical entry is discarded. If all (obligatory) nodes in the RLCS match against nodes in the CLCS, then the rest of the CLCS is recursively checked against other lexical entries stored at the remaining unmatched CLCS nodes. Some nodes, indicated with a "*", as in Figure 2, require not just a match against the corresponding CLCS node, but also a match against another lexical entry. Some CLCS nodes must thus match multiple RLCS nodes. A CLCS node matches an RLCS node, if the following conditions hold:

(7) a. the primitives are the same (or primitive for one is a wild-card, represented as `nil`)

b. the types (e.g., thing, event, state, etc.) are the same

c. the fields (e.g., identificational, possessive, locational, etc) are the same

d. the positions (e.g., subject, argument, or modifier) are the same

e. all obligatory children of the RLCS node have corresponding matches to children of the CLCS

Subject and argument children are obligatory unless specified as optional, whereas modifiers are optional unless specified as obligatory. In the RLCS for ``reduce'' in Figure 2, the nodes corresponding to agent and theme (numbered 1 and 2, respectively) are obligatory, while the instrument (the node numbered 19) is optional. Thus, even though there is no matching lexical entry for node 20 ("*"-marked in the RLCS for "with"), the main RLCS for ``reduce'' is allowed to match, though without any realization for the instrument.

A complexity in the algorithm occurs when there are multiple possibilities filling in a position in a CLCS. In this case, only one of these possibilities is required to match all the corresponding RLCS nodes in order for a lexical entry to match. In the case where there are some of these possibilities that do not match any RLCS nodes (meaning there are no target-language realizations for these constructs), these possibilities can be pruned at this stage. On the other hand, ambiguity can also be introduced at the decomposition stage, if multiple lexical entries can match a single structure

The result of the decomposition process is a match-structure indicating the hierarchical relationship between all lexical entries, which, together cover the input CLCS.

3.3.1 LCS-AMR Creation

The match structure resulting from decomposition is then converted into the appropriate input format used by the Nitrogen generation system. Nitrogen's input, Abstract Meaning Representation (AMR), is a labeled directed graph written using the syntax for the PENMAN Sentence Plan Language (Penman 1989). the structure of an AMR is basically as in (8).

(8) AMR = <concept> | (<label> {<role> <AMR>}+)

Since the roles expected by Nitrogen's English generation grammar do not match well with the thematic roles and features of a CLCS, we have extended the AMR language with LCS-specific relations, calling the result an, an LCS-AMR. To distinguish the LCS relations from those used by Nitrogen, we mark most of the new roles with the prefix `:LCS-`. Figure 3 shows the LCS-AMR corresponding to the CLCS in (4).

```
(a7537 / |reduce|
    :LCS-NODE 6253520
    :LCS-VOICE ACTIVE
    :CAT V
    :TELIC +
    :LCS-AG (a7538 / |United States|
        :LCS-NODE 6278216
        :CAT N)
    :LCS-TH (a7539 / |quota|
        :LCS-NODE 6278804
        :CAT N
        :LCS-MOD-THING (a7540 / |china|
            :LCS-NODE 6108872
            :CAT N)
        :LCS-MOD-THING (a7541 / |textile|
            :LCS-NODE 6111224
            :CAT N)
        :LCS-MOD-THING (a7542 / |export|
            :LCS-NODE 6112400
             :CAT N))
    :LCS-MOD-MANNER (a7543 / |unilaterally|
        :LCS-NODE 6279392
        :CAT ADV))
```

Figure 3: LCS-AMR

In the above example, the basic role / is used to specify an instance. So, the LCS-AMR can be read as an instance of the concept |reduce| whose category is a verb and is in the active voice. Moreover, |reduce| has two thematic roles related to it, an agent and a theme; and it is modified by the concept |unilaterally|. The different roles modifying |reduce| come from different origins. The :LCS-NODE value comes directly from the unique node number in the input CLCS. The category, voice and telicity are derived from features of the LCS entry for the verb |reduce| in the English lexicon. The specifications agent and theme come from the LCS representation of the verb reduce in the English lexicon as well, as can be seen by the node numbers 1 and 2, in the lexicon entry in Figure 2. The role :LCS-MOD-MANNER is made up of combining the fact that the corresponding AMR had a modifier role in the CLCS and because its type is a Manner.

3.4 Realization

The LCS-AMR representation is then passed to the realization module. The strategy used by Nitrogen is to over-generate possible sequences of English from the ambiguous or under-specified AMRs and then decide amongst them based on bigram frequency. The interface between the Linearization module and the Statistical Extraction module is a word lattice of possible renderings. The Nitrogen package offers support for both subtasks, Linearization and Statistical Extraction. Initially, we used the Nitrogen grammar to do Linearization. But complexities in recasting the LCS-AMR roles as standard AMR roles as well as efficiency considerations compelled us to create our own English grammar implemented in Lisp to generate the word lattices.

3.4.1 Linearization

In this module, we force linear order on the unordered parts of an LCS-AMR. This is done by recursively calling subroutines that create various phrase types (NP,PP, etc.) from aspects of the LCS-AMR. The result of the linearization phase is a word lattice specifying the sequence of words that make up the resulting sentence and the points of ambiguity where different generation paths are taken. (9) shows the word lattice corresponding to the LCS-AMR in (8).

(9) (SEQ (WRD "*start-sentence*" BOS) (WRD "united states" NOUN) (WRD "unilaterally" ADJ) (WRD "reduced" VERB) (OR (WRD "the" ART) (WRD "a" ART) (WRD "an" ART)) (WRD "china" ADJ) (OR (SEQ (WRD "export" ADJ) (WRD "textile" ADJ)) (SEQ (WRD "textile" ADJ) (WRD "export" ADJ))) (WRD "quota" NOUN) (WRD "." PUNC) (WRD "*end-sentence*" EOS))

The keyword SEQ specifies that what follows it is a list of words in their correct linear order. The keyword OR specifies the existence of different paths for generation. In the above example, the word 'quota' gets all possible determiners since its definiteness is not specified. Also, the relative order of the words 'textile' and 'export' is not resolved so both possibilities are generated.

Sentences were realized according to the pattern in (10). That is, first subordinating conjunctions, if any, then modifiers in the temporal field (e.g., "now", "in 1978"), then the first thematic role, then most other modifiers, the verb (with collocations if any) then spatial modifiers ("up", "down"), then the second and third thematic roles, followed by prepositional phrases and relative sentences. Nitrogen's morphology component was also used, e.g., to give tense to the head verb. In the example above, since there was no tense specified in the input LCS, past tense was used on the basis of the telicity of the verb.

(10) (Sconj ,) (temp-mod)* Th1 (Mods)* V (coll) (smod)* (Th2)+ (Th3)+ (PP)* (RelS)*

There is no one-to-one mapping between a particular thematic role and an argument position. For example, a theme can be the subject in some cases and it can be the object in others or even an oblique. Observe "cookie" in (11).

(11) a. John ate a cookie (object)
b. the cookie contains chocolate (subject)
c. she nibbled at a cookie (oblique)

Thematic roles are numbered for their correct realization order, according to the hierarchy for arguments shown in (12).

(12) *agent > instrument > theme > perceived > (everythingelse)*

So, in the case of the occurrence of theme alone, it is mapped to first argument position. If a theme and an agent occur, the agent is mapped to first argument position and the theme is mapped to second argument position. A more detailed discussion is available in (Dorr et al., 1998). For the LCS-AMR in Figure 3, the thematic hierarchy is what determined that the |united states| is the subject and |quota| is the object of the verb |reduce|.

In our input CLCSs, in most cases little hierarchical information was given about multiple modifiers of a noun. Our initial, brute force, solution was to generate all permutations and depend on statistical extraction to decide. This technique worked for noun phrases of about 6 words, but was too costly for larger phrases (of which there were several examples in our test corpus). This cost was alleviated to some degree, also providing slightly better results than pure bigram selection by ordering adjectives within classes, inspired by the adjective ordering scheme in (Quirk et al., 1985). This is shown in (13).

(13) a. Determiner (all, few, several, some, etc.)
b. Most Adjectival (important, practical, economic, etc.)
c. Age (old, young, etc.)
d. Color (black, red, etc.)
e. Participle (confusing, adjusted, convincing, decided)

f. Provenance (China, southern, etc.)

g. Noun (Bank_of_China, difference, memorandum, etc.)

h. Denominal (nouns made into adjectives by adding -al, e.g., individual, coastal, annual, etc.)

If multiple words fall within the same group, permutations are generated for them. This situation can be seen for the LCA-AMR in Figure 3 with the ordering of the modifiers of the word |quota|: |china|, |export| and |textile|. |china| fell within the Provenance class of modifiers which gives it precedence over the other two words. They, on the other hand, fell in the Noun class and therefore both permutations were passed on to the statistical component.

3.4.2 Statistical Preferences

The final step, extracting a preferred sentence from the word lattice of possibilities is done using Nitrogen's Statistical Extractor without any changes. Sentences are scored using uni and bigram frequencies calculated based on two years of Wall Street Journal (Langkilde and Knight, 1998b).

4 Dealing with Ambiguity

A major issue in sentence generation from an interlingua or conceptual structure, especially as part of a machine translation project, is how and when to deal with ambiguity. There are several different sources of ambiguity in the generation process outlined in the previous section. Some of these include:

- ambiguity in source language analysis (as represented by *possibles* nodes in the CLCS input to the Generation system). This can include ambiguity between multiple concepts, such as the example in (5), LCS type/structure (e.g., thing or event, which field), or structural ambiguity (subject, argument or modifier).
- ambiguity introduced in lexical choice (when multiple match structures can cover a single CLCS)
- ambiguity introduced in realization (when multiple orderings are possible, also multiple morphological realizations)

There are also several types of strategies for addressing ambiguity at various phases, including:

- passing all possible structures down for further processing stages to deal with
- filtering based on "soft" preferences (only pass the highest set of candidates, according to some metric)
- quota-based filtering, passing only the top n candidates
- threshold filtering, passing only candidates that exceed a fixed threshold (either score or binary test)

The generation system uses a combination of these strategies, at different phases in the processing. Ambiguous CLCS sub-trees are sometimes annotated with scores based on preference of attachment as an argument rather than a modifier. The alignment algorithm can be run in either of two modes, one which selects only the top scoring possibility for which a matching structure can be found, and one in which all possible structures are passed on, regardless of score. The former method is the only one feasible when given very large (e.g., over 1 megabyte text files) CLCS inputs. Also at the decomposition level, soft preferences are used in that missing lexical entries can be hypothesized to cover parts of the CLCS (essentially "making up" words in the target language). This is done, however, only when no legitimate matches are found using only the available lexical entries. At the linearization phase, there are often many choices for ordering of modifiers at the same level. As mentioned in the previous section, we are experimenting with separating these into positional classes, but our last resort is to pass along all permutations of elements in each sub-class. The ultimate arbiter is the statistical extractor, which orders and presents the top scoring realizations.

5 Interlingual representation issues

One issue that needs to be confronted in an Interlingua such as LCS is what to do when linguistic structure of languages vary widely, and useful conceptual structure may also diverge from these. A case in point is the representation of numbers. Languages diverge widely as to which numbers are primitive terms, and how larger numbers are built compositionally through modification (e.g., multiplication and addition). One question that immediately comes up is whether an interlingua such as LCS should represent numbers according to the linguistic structure of the source language (or some particular designated natural language) or as some other internal numerical form, (e.g. decimal numerals). Likewise, on generation into a target language, how much of the structure of the source language should be kept, especially when this is not the most natural way to group things in the target language. One might be tempted to always convert to a standard interlingua representation of numbers, however this does lose some possible classification into groups that might be present in the input (contrast in English: "12 pair" with "2 dozen".

In our Chinese-English efforts, such issues came up, since the natural multiplication points in Chinese were 100, 10,000, and 100,000,000, rather than

100, 1000, and 1,000,000, as in English. Our provisional solution is to propogate the source language modification structure all the way through the LCS-AMR stage, and include special purpose rules looking for the "Chinese" numbers and multiplying them together to get numerals, and then divide and realize in the English fashion. E.g., using the words thousand, million, and billion.

6 Evaluation

So far most of the evaluation has been fairly small-scale and fairly subjective, generating English sentences from CLCSs produced from about 80 sentences. Evaluation in this case is difficult, because the ultimate criteria is translation quality, which can, itself, be difficult to judge, but, moreover, it can be hard to attribute specific deficits to the analysis phase, the lexical resources, or the generation system proper. So far results have been mostly adequate, even for large and fairly complex sentences, taking less than 1 minute for generation up to inputs of about 1 megabyte input CLCS files. Ambiguity and complexity beyond that level tends to overtax the generation system.

For the most part, the over-generation strategy of Nitrogen, coupled with the bigram preferences works very well. There are still some difficulties, however. One major one is that, especially with its bias for shorter sentences, fluency is given preference over translation fidelity. Thus, if there are options of whether or not to express some optional information, this will tend to be left out. Also, bigrams are obviously inadequate for capturing long-distance dependencies, and so, if things like agreement are not carefully controlled in the symbolic component, they will be incorrect in some cases.

The generation component has also been used on a broader scale, generating thousands of simple sentences - at least one for each verb in the English LCS lexicon, creating sentence templates to be used in a Cross-Language information retrieval system (Levow et al., 2000).

7 Future Work

The biggest remaining step is a more careful evaluation of different sub-systems and preference strategies to more efficiently process very ambiguous and complex inputs, without substantially sacrificing translation quality. Also a current research topic is how to combine other metrics coming from various points in the generation process with the bigram statistics, to result in better overall outputs.

Another topic of interest is developing other language outputs. Most of the subcomponents are language-independent (assuming of course a different target-language lexicon). The realization components being an obvious exception.

References

Bonnie J. Dorr, M. Antonia Martí, and Irene Castellón. 1997. Spanish EuroWordNet and LCS-Based Interlingual MT. In *Proceedings of the Workshop on Interlinguas in MT, MT Summit, New Mexico State University Technical Report MCCS-97-314*, pages 19–32, San Diego, CA, October.

Bonnie J. Dorr, Nizar Habash, and David Traum. 1998. A Thematic Hierarchy for Efficient Generation from Lexical-Conceptal Structure. In *Proceedings of the Third Conference of the Association for Machine Translation in the Americas, AMTA-98, in Lecture Notes in Artificial Intelligence, 1529*, pages 333–343, Langhorne, PA, October 28–31.

Bonnie J. Dorr. 1992, 1993. The Use of Lexical Semantics in Interlingual Machine Translation. *Machine Translation*, 7(3):135–193.

Bonnie J. Dorr. 1993. *Machine Translation: A View from the Lexicon*. The MIT Press, Cambridge, MA.

Ray Jackendoff. 1983. *Semantics and Cognition*. The MIT Press, Cambridge, MA.

Ray Jackendoff. 1990. *Semantic Structures*. The MIT Press, Cambridge, MA.

Ray Jackendoff. 1996. The Proper Treatment of Measuring Out, Telicity, and Perhaps Even Quantification in English. *Natural Language and Linguistic Theory*, 14:305–354.

Irene Langkilde and Kevin Knight. 1998a. Generation that Exploits Corpus-Based Statistical Knowledge. In *ACL/COLING 98, Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (joint with the 17th International Conference on Computational Linguistics)*, pages 704–710, Montreal, Canada.

Irene Langkilde and Kevin Knight. 1998b. The Practical Value of N-Grams in Generation. In *International Natural Language Generation Workshop*.

Gina-Anne Levow, Bonnie J. Dorr, and Maria Katsova. 2000. Construction of Chinese-English Semantic Hierarchy for Cross-Language Retrieval. In *Proceedings of the Workshop on English-Chinese Cross Language Information Retrieval, International Conference on Chinese Language Computing*, pages 187–194, Chicago, IL.

R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman.