Machine Translation, 12:4, 271–325 (1997)

Large-Scale Dictionary Construction for Foreign Language Tutoring and Interlingual Machine Translation

BONNIE J. DORR bonnie@cs.umd.edu
Department of Computer Science and UMIACS, University of Maryland, College Park, MD 20742

Received October 15, 1995; Revised March 15, 1997

Abstract. This paper describes techniques for automatic construction of dictionaries for use in large-scale foreign language tutoring (FLT) and interlingual machine translation (MT) systems. The dictionaries are based on a language-independent representation called *lexical conceptual structure* (LCS). A primary goal of the LCS research is to demonstrate that synonymous verb senses share distributional patterns. In this paper, we show how the syntax-semantics relation can be used to develop a lexical acquisition approach that contributes both toward the enrichment of existing online resources and toward the development of lexicons containing more complete information than is provided in any of these resources alone. We start by describing the structure of the LCS and showing how this representation is used in FLT and MT. We then focus on the problem of building LCS dictionaries for large-scale FLT and MT. First, we describe authoring tools for manual and semi-automatic construction of LCS dictionaries; we then present a more sophisticated approach that uses linguistic techniques for building word definitions automatically. These techniques have been implemented as part of a set of lexicon-development tools used in the MILT FLT project (Dorr et al., 1995; Sams, 1995; Weinberg et al., 1995) and in the PRINCITRAN MT project (Dorr et al., 1995b).

Keywords: lexical acquisition, foreign language tutoring, interlingual MT, semantic verb classes, syntactic codes

1. Introduction

Current advances in lexical semantics make it possible to provide a wide range of new capabilities in natural language processing (NLP) systems. In particular, recent work in computational linguistics (Carrier and Randall, 1993; Dorr, 1993; Dorr, 1994; Fillmore, 1968; Foley and Van Valin, 1984; Grimshaw, 1990; Grimshaw, 1994; Gruber, 1965; Hale and Keyser, 1993b; Hale and Keyser, 1993a; Jackendoff, 1983; Jackendoff, 1990; Jackendoff, 1996; Levin, 1993; Levin and Rappaport Hovav, To appear; Olsen, To appear in 1997; Pesetsky, 1982; Pinker, 1989) has suggested that there is a close relation between underlying lexical-semantic structures and their associated syntactic behaviors. This relation can be exploited to develop an interlingual framework that supports activities such as intelligent language tutoring and machine translation. While it is the case that these lexical structures fall short of providing full semantic inferencing, and while knowledge based systems are still limited in the domains for which such inferencing can be supported, these structures

provide a robust basis for the development of language-processing functions and an analysis that is more useful than syntactic analysis.

This paper describes the acquisition of dictionaries based on *lexical conceptual structure* (LCS), a language-independent representation used in the NLP component of implemented foreign language tutoring (FLT) and interlingual machine translation (MT) systems. A primary goal of the LCS research is to demonstrate that synonymous verb senses share distributional patterns. In this paper, we show how the syntax-semantics relation can be used to develop a lexical acquisition approach that contributes both toward the enrichment of existing online resources and toward the development of lexicons containing more complete information than is provided in any of these resources alone.

The next section describes the structure of the LCS. Section 3 shows how this representation is used to support a question-answering component of a FLT system called MILT (Dorr et al., 1995; Sams, 1995; Weinberg et al., 1995) and to resolve cross-linguistic divergences in an interlingual MT system called PRINCITRAN (Dorr et al., 1995b). The remaining sections focus on the problem of large-scale, computer-aided acquisition of LCS dictionaries for these two applications. Section 4 describes authoring tools for manual and semi-automatic construction of the LCS representation. Section 5 presents a more sophisticated approach that uses linguistic techniques for building word definitions automatically. These techniques have been implemented as part of a set of lexicon-development tools used in the MILT and PRINCITRAN projects.

2. Lexical Conceptual Structure

One of the types of knowledge that must be captured in FLT and MT is linguistic knowledge at the level of the lexicon, which covers a wide range of information types, such as verbal subcategorization for events (e.g., that a transitive verb such as "hit" occurs with an object noun phrase), featural information (e.g., that the direct object of a verb such as "frighten" is animate), thematic information (e.g., that "John" is the agent in "John hit the ball"), and lexical-semantic information (e.g., that spatial verbs such as "throw" are conceptually distinct from verbs of possession such as "give"). By modularizing the lexicon, we treat each information type separately, thus allowing us to vary the degree of dependence on each level, so that we can address the question of how much knowledge is necessary for the success of the particular NLP application.

The most intricate component of lexical knowledge is the lexical-semantic information, which is encoded in the form of Lexical Conceptual Structure (LCS) as formulated by Dorr (1993; 1994) based on work by Jackendoff (1983; 1990). This representation abstracts away from syntax just far enough to enable language independent encoding, while retaining enough structure to be sensitive to the requirements for multi-lingual processing. The central idea of LCS is that human language can be modeled using a uniform internal representation of conceptual information with a few parameters that can be toggled one way or another to accommodate vari-

ous languages. The versatility of the LCS allows us to reuse labor-intensive features of the lexicon across languages. Natural language analysis is performed by mapping the syntactic representation of the input sentences into this internal representation; the result is used later as the basis for semantic interpretation. This technique produces more accurate analyses than naive syntactic-tree matching approaches, which tend to be much more limited in coverage.

The LCS approach views semantic representation as a subset of conceptual structure, the language of mental representation, as in (Jackendoff, 1983; Jackendoff, 1990). This approach includes *types* such as Event and State, which are specialized into *primitives* such as GO, STAY, BE, GO-EXT, and ORIENT. As an example of how the primitive GO is used to represent sentence semantics, consider the LCS for *John jogged to school*:

(1) $[_{\text{Event}}\ \text{GO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{JOHN}],$
$[_{\text{Path}}\ \text{TO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{JOHN}],$
$[_{\text{Position}}\ \text{AT}_{\text{Loc}}\ ([_{\text{Thing}}\ \text{JOHN}], [_{\text{Thing}}\ \text{SCHOOL}])])])]$

To this representation we add a manner component $[_{\text{Manner}}$ JOGGINGLY] to distinguish among verbs, e.g. *run*, *walk*, and *jog*. We also employ Jackendoff's notion of *field*, which carries Loc(ational) semantic primitives into non-spatial domains such as Poss(essional), Temp(oral), Ident(ificational), Circ(umstantial), and Exist(ential). The full representation for (1) is therefore (2), roughly 'John went to the school by jogging.'

(2) $[_{\text{Event}}\ \text{GO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{JOHN}],$
$[_{\text{Path}}\ \text{TO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{JOHN}],$
$[_{\text{Position}}\ \text{AT}_{\text{Loc}}\ ([_{\text{Thing}}\ \text{JOHN}], [_{\text{Thing}}\ \text{SCHOOL}])])],$
$[_{\text{Manner}}\ \text{JOGGINGLY}])]$

The LCS captures those aspects of lexical knowledge related to argument structure, not "deeper" notions of meaning such as aspectual, contextual, domain, and world knowledge. Further details about the structure of the LCS representation are given in (Dorr, 1992), and discussion about the role of aspect in the LCS is provided in (Dorr and Olsen, 1996). The next section demonstrates the benefits of using the LCS representation for large-scale NLP applications.

3. Application of the LCS Representation

This section describes the use of the LCS representation in a question-answering component of the MILT FLT system (Dorr et al., 1995; Sams, 1995; Weinberg et

al., 1995) and in cross-linguistic divergence resolution in the PRINCITRAN MT system (Dorr et al., 1995b).

3.1. Foreign Language Tutoring

The MILT tutoring project was developed jointly by the University of Maryland and Micro Analysis and Design Corporation under sponsorship from the Army Research Institute. A primary objective of this project is to allow students to practice language comprehension and production through the use of NLP technology. To support this goal, we have developed a system which encodes a wide range of lexical and semantic knowledge relevant for understanding utterances.[1] The system includes a number of authoring tools, speech and written display lessons, and various types of language drills. The most NLP-intensive components of the system include an authorable question-answering lesson and a limited-domain dialog lesson; we focus on the first of these two lesson types.

The LCS representation is used as the basis of matching routines for assessing students' answers to free response questions about a short foreign language passage. In order to inform the student whether a question has been answered correctly, the instructor (also called the 'author') of the lesson must provide the desired response in advance. The system parses and semantically analyzes the author's response into a corresponding LCS representation which is then prestored in a database of possible responses. Once the question answering lesson is activated, each of the student's responses is parsed and semantically analyzed into a LCS representation which is checked for how closely the student's response LCS matches the author's prestored LCS. The student is then informed as to whether the question has been answered correctly.

The overall processing design used to support question answering exercises is shown in Figure 1. The analysis component of the system includes morphological, syntactic, and semantic processing that transforms the student's input sentence (i.e., the response to the tutor's question) into a language-independent representation. The syntactic analyzer makes use of Government Binding (GB) principles (developed by Chomsky (1981; 1986) and his followers) that are parameterized for the particular language in question. Thus, the analyzer is able to function in both Arabic and Spanish, given the appropriate lexicon. The parse tree produced by the syntactic analyzer is passed to the semantic analysis/composition component, and the language-independent LCS representation is generated using parameterized mapping rules. This representation is then passed to the LCS Matcher which checks the answer for a match against the desired answer that was prestored by the author. The degree to which the match succeeds is then determined, and this is used to generate the tutor output (i.e., confirmation of (in)correctness).

A brief example is given here. (See (Dorr et al., 1995) for more details.) Consider what happens in a lesson if the author has specified that a correct answer in Spanish is *John corrió a la casa* ('John ran to the house'). This is processed by the system to produce the following LCS:

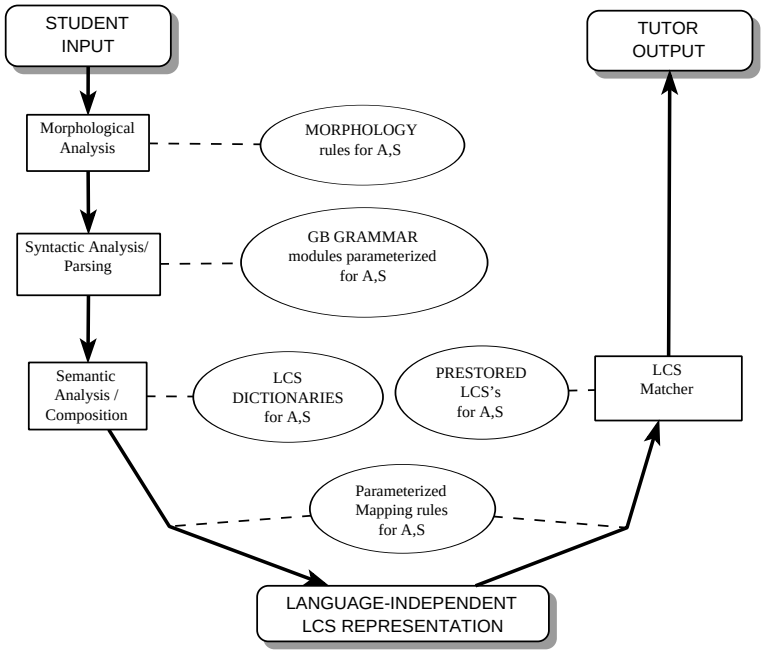

Figure 1. Diagram of Processing Flow for Question/Answering in MILT

(3) [$_{\text{Event}}$ GO_{Loc}
([$_{\text{Thing}}$ JOHN],
[$_{\text{Path}}$ TO_{Loc}
([$_{\text{Position}}$ AT_{Loc} ([$_{\text{Thing}}$ JOHN], [$_{\text{Thing}}$ HOUSE])])],
[$_{\text{Manner}}$ RUNNINGLY])]

This is stored by the tutor and then later matched against the student's answer. If the student types *John fue a la casa* ('John went to the house'), the system must determine if these two match. The student's sentence is processed and the following LCS structure is produced:

(4) [$_{\text{Event}}$ GO_{Loc}
([$_{\text{Thing}}$ JOHN],
[$_{\text{Path}}$ TO_{Loc}
([$_{\text{Position}}$ AT_{Loc} ([$_{\text{Thing}}$ JOHN], [$_{\text{Thing}}$ HOUSE])])])]

The matcher compares these two, and produces the following output:

```
Missing :
 MANNER RUNNINGLY

Extra :
 NIL

INCORRECT answer.
```

This identifies the student's response as an incorrect answer, since the student has omitted the fact that John was running.

If we reverse the situation, with the author storing as the correct answer *John fue a la casa* ('John went to the house') and the student typing *John corrió a la casa* ('John ran to the house'), the situation would be different. In this case, the matcher would return the following:

```
Missing :
 NIL

Extra :
 MANNER RUNNINGLY

CORRECT answer.
```

This time the student has provided information the author did not require, but omitted nothing that was required. Thus, the system has identified the answer as correct and recognized that the student provided information about the manner of the going, which was not necessary.

The above output—provided by the NLP component of the Tutor—is not the final response displayed to the student. The system must convert this information into meaningful feedback so that the student knows how to repair the answer. Since the focus of this paper is on the LCS representation and means of automatic

acquisition for large-scale systems, we will not describe the full range of feedback the tutor could potentially provide for each NLP output. Some possibilities are summarized (in English) in Table 1 (adapted from (Holland, 1994)).

Table 1. Correspondence Between NLP Output and Tutor Feedback

System Prompt: What did John do?			
Student Answer	**Prestored Answer**	**Matcher Output**	**Feedback**
John ran home John raced to his house, etc.	John ran home	exact match	"That's right"
John is nice	John ran home	mismatch primitive	"Please reread"
John went home	John ran home	missing MANNER	"How?"
John ran	John ran home	missing argument	"Where?"
John ran home	John went home	extra MANNER	"You're assuming things"
John stabbed Jill	Jill stabbed John	mismatch filler	"Right idea, missed details"

One advantage of the LCS matching becomes clear in the examples given in this table. It would be a tedious task if the author were expected to specify, in advance, all possible ways that the student might choose to convey the desired information. Our approach to representing the prestored answer allows the author to type in an answer that is general enough to match any number of additional answers. For example, the prestored answer *John ran home* corresponds to several correct answers including *John raced home*, *John dashed home*, etc.

Another advantage concerns the handling of inappropriate answers. If the students types a completely inappropriate answer, such as *Me llamo John* ('I am called John') the corresponding LCS would be:

(5) $[_{\text{Event}}\ \text{BE}_{\text{Ident}}\ ([_{\text{Thing}}\ \text{I}], [_{\text{Position}}\ \text{AT}_{\text{Ident}}\ ([_{\text{Thing}}\ \text{I}], [_{\text{Property}}\ \text{JOHN}])])]$

This is identified by the matcher as being in the wrong primitive class, and rejected as an inappropriate answer. The tutor informs the student that the correct answer is expressed differently and the student tries again.

It should be noted that the lack of a more complex knowledge representation system underlying the matching makes for some limitations to this technique. For example, consider the following student response to the question as to what happened to John:

STUDENT> Carlos corrió detrás de John hasta que llegaron a la casa.
(Carlos ran after John until they both reached the house.)

In this case, the system would generate the following LCS:

(6) $[_{\text{Event}}\ \text{GO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{CARLOS}],$
$[_{\text{Path}}\ \text{TOWARD}_{\text{Loc}}$
$([_{\text{Position}}\ \text{BEHIND}_{\text{Loc}}\ ([_{\text{Thing}}\ \text{CARLOS}], [_{\text{Thing}}\ \text{JOHN}])])],$
$[_{\text{Manner}}\ \text{RUNNINGLY}],$
$[_{\text{Position}}\ \text{UNTIL}_{\text{Temp}}$

(*HEAD*,
[$_{\text{State}}$ BE$_{\text{Loc}}$
([$_{\text{Thing}}$ REFERENT],
[$_{\text{Position}}$ AT$_{\text{Loc}}$ ([$_{\text{Thing}}$ REFERENT], [$_{\text{Thing}}$ HOUSE])])])])

This would not match the LCS generated previously for 'John ran home.' The system would need to infer that running after John (until the house is reached) implies that John ran to the house. This is beyond the scope of the LCS system. (See (Dorr et al., 1995) for discussion about the addition of such a capability.)

In addition to question answering, the usefulness of the LCS framework and the matching algorithm extends to the Text Translation lesson type in the MILT system, in which the student must translate an English sentence into a sentence in either Arabic or Spanish. Again, the author provides a sample question (this time the sentence to be translated) and answer. The answer is parsed, analyzed and stored as described above. The student's translation and the prestored answer are matched, and the tutor is able to analyze whether the translation is correct or not.

3.2. Interlingual Machine Translation

The PRINCITRAN translation project is currently under development at the University of Maryland under sponsorship from the Army Research Office and Battelle Corporation. The domain of the system is military messages in Korean, French, Spanish, and English. As in the MILT system, the LCS representation serves as the basis of this system. The overall processing design used to support interlingual MT is shown in Figure 2.

The diagram for Interlingual MT is analogous to that of the FLT system in a number of ways: the analysis component of the system includes morphological, syntactic, and semantic processing that transforms the source-language input into a language-independent representation. As above, the parse tree produced by the syntactic analyzer is passed to the semantic analysis/composition component, and the language-independent LCS representation is produced. The LCS is then used by the synthesis component, which includes syntactic, semantic, and morphological modules that mirror the modules of the analysis component.

Some of the most difficult phenomena handled in PRINCITRAN rely heavily on how the LCS representations are encoded. The LCSs in the lexicon are combined through a composition process which produces an interlingua for the source- and target-language sentence. Consider the following example:

(7) E: I like Mary
S: María me gusta
(Mary (to) me pleases)

The interlingua that is produced by the composition process is the following:

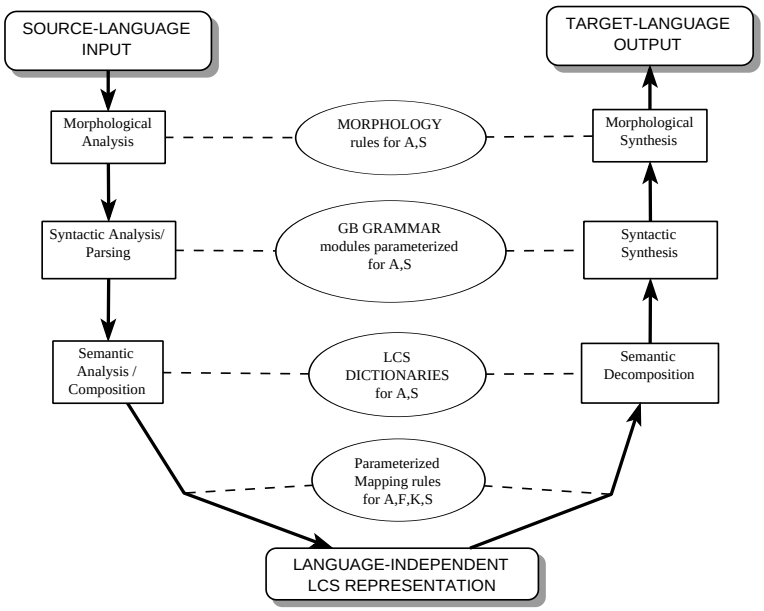

Figure 2. Diagram of Processing Flow in PRINCITRAN

(8) $[_{\text{State}}$ BE_{Ident} ($[_{\text{Thing}}$ I],
$[_{\text{Position}}$ AT_{Ident} ($[_{\text{Thing}}$ I], $[_{\text{Thing}}$ MARY])],
$[_{\text{Manner}}$ LIKINGLY])]

This representation roughly means "I am in an identificational state LIKINGLY with respect to Mary." Both the Spanish and English sentences are based on this representation. To account for the syntactic distinction above (the subject-object reversal) we encode the main verb as a parameterized LCS in the lexicon:

(9) (i) **Lexical Entry for like**:
$[_{\text{State}}$ BE_{Ident} ($[_{\text{Thing}}$:EXT W],
$[_{\text{Position}}$ AT_{Ident} ($[_{\text{Thing}}$ W], $[_{\text{Thing}}$:INT Z])],
$[_{\text{Manner}}$ LIKINGLY])]

(ii) **Lexical Entry for gustar**:
$[_{\text{State}}$ BE_{Ident} ($[_{\text{Thing}}$:INT W],
$[_{\text{Position}}$ AT_{Ident} ($[_{\text{Thing}}$ W], $[_{\text{Thing}}$:EXT Z])],
$[_{\text{Manner}}$ LIKINGLY])]

The :INT/:EXT marker is one case of lexical parameterization that allows the system to account for cross-linguistic divergences during the LCS composition process. The full range of distinctions that are resolved by means of lexical parameterization is described in (Dorr, 1994).

4. Building a LCS Lexicon: Authoring Tools

We now demonstrate the feasibility of using the LCS representation in large-scale NLP applications such as those described above. In particular, we show that, if we rely on constraints provided by current linguistic theory, techniques for automatic acquisition of these representations is now well within our grasp.

Our goal is to construct LCS lexicons automatically for English, Arabic, Korean, Spanish, and French. A progression of dictionary-construction techniques are available, ranging from a LCS Editor to a more advanced semantic classification tool that feeds into an LCS construction program called LEXICALL. Table 2 illustrates this range.

At one end of the lexicon-tool spectrum (Manual), the level of knowledge assumed by the dictionary builder must be relatively extensive, the level of tedium high, and development time lengthy; however, linguistic coverage is easily controlled by the builder, making full domain-specific coverage possible. At the other end of the spectrum (Automatic), there need not be prior linguistic knowledge and development is automatic (i.e., not tedious or lengthy); however, the lexicon coverage is likely to have "holes" in application-specific information due to the fact that the content of machine-readable resources cannot always be predicted in advance.

The majority of previous approaches to lexical acquisition for NLP applications has fallen into the first two categories, i.e., Manual and Semi-automatic. In manual approaches, the user directly manipulates the underlying (machine-readable) representations associated with word definitions. By contrast, semi-automatic tools

Table 2. Spectrum of Automation Possibilities: Lexicon Building for NLP Applications

Spectrum	**Lexicon Tool**
Manual	Linguist's LCS Editor (Dorr et al., 1995)
Semi-Automatic	TEAM (Grosz et al., 1987) IRUS (Bates and Bobrow, 1983) TELI (Ballard and Stumberger, 1986) LUKE (Knight, 1991) User's LCS Editor (Dorr, this article) Others: Ginsparg (1983), Guida and Tasso (1983), Grishman (1986), Thompson and Thompson (1983), Templeton and Burger (1983), etc.
Automatic	BICORD (Klavans and Tzoukermann, 1995) SILC (Wu and Xia, 1995) LUCIFER (Copestake et al., 1995) Lexicon Creation Program (Lonsdale, Mitamura, and Nyberg, 1995) Acquisition Using Semantic Filters (Dorr and Jones (1996a; 1996b)) Automatic Semantic Classification (Dorr, this article) LEXICALL (Dorr, To appear in 1997) Others: Church and Hanks (1990)

allow the user to enter words into the lexicon without direct manipulation of the underlying representation. This section focuses on our own tools developed within these categories, i.e., interactive authoring tools used for construction of LCS-based lexicons.

4.1. Manual Authoring: Linguist's LCS Editor

The first tool we developed was a window-based Linguist's LCS Editor (Dorr et al., 1995) as shown in Figure 3. This editor was initially used to enter vocabulary into the dictionaries for Spanish and Arabic in the MILT system. The idea is to provide an interface that allows LCSs to be built and modified in different languages. During editing, the current state of the LCS is continuously updated and displayed in two windows, the "Tree Representation" window and the "Text Representation" window. In the example shown here, the user is building an LCS representation for the Arabic word *layos* which corresponds to the "Locational" version of the English word *be*.

At the top of the Editor window is a browser which allows the author to set the Type, Primitive, and Field of the selected (highlighted) node in the current LCS. The selected Type determines which Primitives are available, and the selected Primitive determines the Fields that are allowed. Some Primitives have no allowed Fields. The Editor only allows valid choices to be made.

The Editor also displays sample sentences for various combinations of Primitive and Field, to guide the author with respect to the meanings of the various terms. In Figure 3, the words *The statue is in the park* represent an example of an English

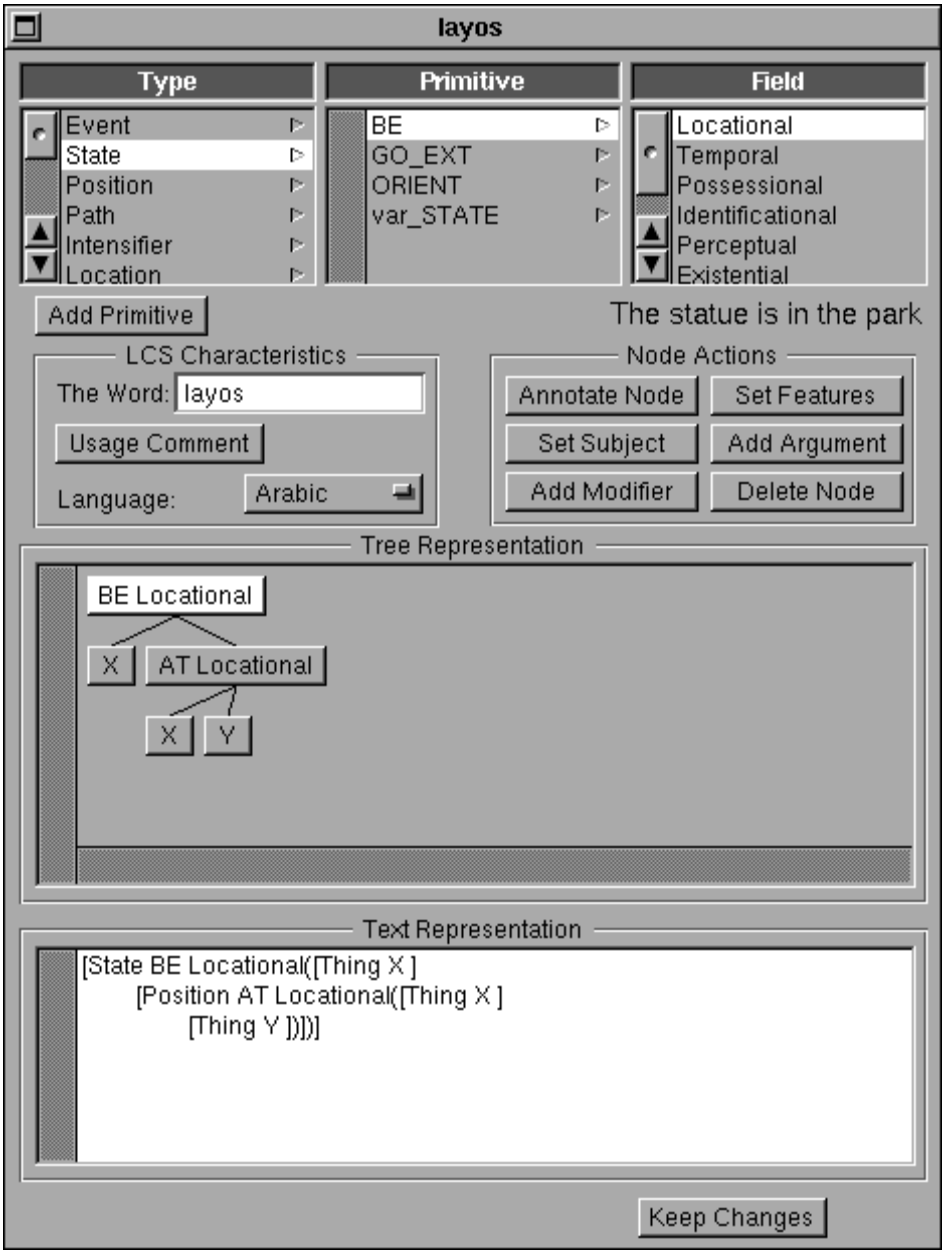

Figure 3. LCS Entry Window of Linguist's LCS Editor

sentence using a verb (e.g., *be*) that has the same Type, Primitive, and Field as the currently-selected node.

Under the browser is the Add Primitive button. Clicking this button allows the author to add a new primitive to the set that is currently defined, as well as specify which Type the primitive belongs to, and which Fields are applicable. The LCS of this example contains nodes labeled X and Y. There are no Primitives by these names; rather, these nodes represent variables. Any nodes with the same variable name will eventually refer to the same item in the natural language sentence that this LCS is used to represent.

The next item down is a group of controls labeled "LCS Characteristics." These are items that apply to the LCS entry as a whole. The author enters the word that the entry represents in the Text Field labeled "The Word." The Usage Comment button allows the author to enter a comment about the usage of this particular entry for documentation purposes. (It has no effect on the correctness of the entry.) Below this button is a language button labeled "Arabic," reflecting the fact that this is the LCS entry for the Arabic verb, *layos*.

To the right of the "LCS Characteristics" group is another group of controls labeled "Node Actions." These items apply solely to the node that is currently selected. In the current example, the root node of the tree (i.e., Be Locational) is selected, so any of the "Node Actions" items that are chosen would apply to the root node. The selection of a node action would open up a secondary window (palette) that supports the annotation of LCS items with classifying markers (such as the :INT/:EXT markers given in example (9) earlier), semantic and syntactic feature information, and usage comments. If the tree grew large enough, vertical and/or horizontal scrollers would appear to allow the author to see different regions of the tree.

While the Linguist's LCS Editor is useful, it is virtually non-automatic. Much of the tedium of the dictionary-construction process has not been eliminated. The Editor requires the user to have a great deal of linguistic knowledge because the LCS is a more sophisticated encoding of knowledge than that of other representations typically found in NLP lexicons (e.g., syntactic subcategorization). Nevertheless, the Editor *does* provide a constraint satisfaction mechanism which allows certain types of information to be filled in automatically while the user is building the representation. This is where the benefit of the LCS representation is most apparent: the LCS conforms to well-formedness conditions that allow certain types of information to be derived automatically without the requirement that the user explicitly state these constraints. An example of such a constraint is the fact that the primitive GO always takes two arguments, a Thing and a Path. The use of these and additional constraints allow us to head toward the semi-automatic version of this tool to be described next.

4.2. Semi-Automatic Authoring: User's LCS Editor

Many lexicon development tools fall under the category of "Semi-Automatic:" they require human intervention but are designed for users with no specialized training in linguistics. Most tools in this category still look much like the first such interface created for the TEAM project (Grosz et al., 1987). Others who have worked on such lexical interfaces are Bates and Bobrow (1983), Ballard and Stumberger (1986), Knight (1991), Ginsparg (1983), Guida and Tasso (1983), Grishman (1986), Thompson and Thompson (1983), and Templeton and Burger (1983). The user is presented with various sentences utilizing the word being updated and asked to indicate which usages are correct and which are not. Each sentence represents a test to determine whether or not a particular linguistic feature applies.

One problem with these interfaces is that asking these types of questions does not work for all words. Someone with linguistic expertise will still have to review the results of the maintenance session. For example, once the Spanish verb *gustar* is entered into the lexicon as a psych verb, someone with knowledge about linguistic structure must check that the argument structure (i.e., ordering of subject with respect to the object) is the reverse of the argument structure for the analogous English word *like*. (Example (7) given earlier illustrates this thematic distinction between Spanish and English.)

One of the most sophisticated tools in the Semi-Automatic category is the LUKE system by Knight (1991). This system comprises the lexical acquisition component of large NLP system called KBNL (Barnett et al., 1990). LUKE is connected to a knowledge-representation system called CYC (Lenat and Guha, 1990), a large-scale, common-sense, knowledge-based (KB) project started in 1984. The drawback of this system is that the person who does the editing must be a KB expert and, in particular, must know about concepts available in the CYC system. For example, in order to enter the word *eat* into the system, the lexicon builder must first associate the verb *eat* with the concept EatingEvent, which is defined in the CYC ontology. Once this is done, the rest of the process is virtually automatic: a window pops up asking the user to edit the verb *eat* (see Figure 4).

Three types of information may then be entered by the lexicon builder. The first is morphological information. The system makes an educated guess (e.g., that the past tense of *eat* is *eated*) which can then be corrected by the KB expert (e.g., that *eated* should actually be *ate*). Note that the user's input here could easily be replaced by information from a comprehensive machine-readable dictionary.

The second type of information is syntactic: the system guesses at a subcategorization frame by looking at conceptual information. For example, if a concept has an agent and an object (as in the EatingEvent case), then agent-verb-object is assigned to the verb's lexical entry. (This is essentially a basic case frame analysis.) This information is then used to generate sentences about which the user can make grammaticality judgments. For example, the system guesses that *fork* is an instrument of an EatingEvent and *pea* is an object of an EatingEvent, purely because it knows certain properties about forks, peas, and the event of eating. From this

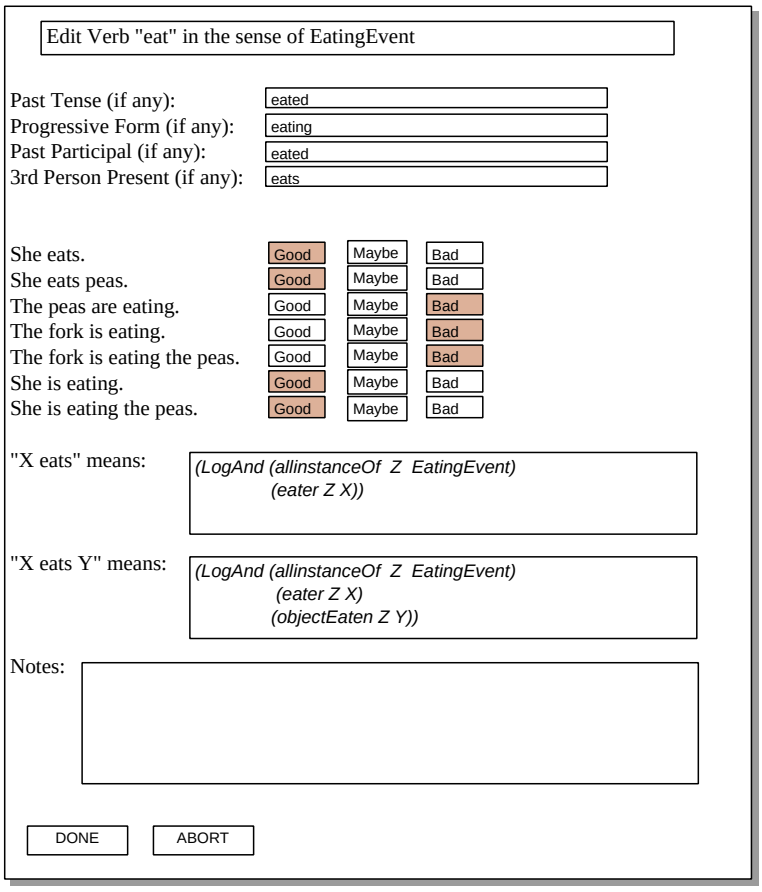

Figure 4. Semi-Automatic Tool: LUKE

information, it produces semantically plausible sentences, e.g., *She eats peas*). The user then makes grammaticality judgments (Good, Maybe, or Bad) for the sentences produced by the system; from these, the system identifies and stores away a set of syntactic frames for the verb.

The system then produces a set of logical restrictions—the third type of information entered into the lexicon—by mapping from syntactic constituents to the semantic roles of the verb. Like the other two types of information, these restrictions may also be modified by the lexicon builder.

Although many of the ideas used in LUKE have been borrowed for our own automatic dictionary construction tools (e.g., the mapping from syntactic constituents to semantic roles), we have found that it is possible to avoid certain drawbacks of the LUKE approach. In particular, LUKE assumes that there is a large knowledge base (in particular, CYC) as well as a person who is entirely familiar with it. Nevertheless, this approach is clearly a step in the right direction toward automation in that it is able to establish morphological and case-frame information and to pose easy-to-read questions about grammaticality, thus producing a linking to semantics on a per-item basis.

Our own tool, designed for semi-automatic construction of LCS representations, allows a non-linguist e.g., a "User" of the Foreign Language Tutoring system—generally assumed to be a foreign language instructor—to enter words into the lexicon without being forced to manipulate the LCS representation directly. The system has an advantage over LUKE in that it does not require a large knowledge base (or a KB expert). We call this tool the User's LCS Editor (as opposed to the Linguist's LCS Editor described in the last section). Many of the design features of the LUKE system carry over into this system, but the goal is to build LCSs (i.e., word meaning and structure) rather than general world knowledge. Figure 5 illustrates a sample session in which the user has constructed a lexical representation for the word *enter*. The system is menu-driven and relies on a sequence of responses to questions that are relevant to semantic components in the LCS representation.

While the User's Editor does not require as much linguistic knowledge as the Linguist's Editor, it is still in the semi-automatic category. On the other hand, the lack of a knowledge base makes the knowledge acquisition task even harder. In particular, it is a non-trivial task for the system to determine, from menu-driven questions, what the argument structure of a verb is. Moreover, these menu-driven questions still require some linguistic knowledge, i.e., the "User" must be able to understand questions such as "Expresses Manner?", "What type of Motion?", and "Goal is location?"

In a later version of this same Editor, we developed a mechanism for producing easy-to-read sentences that correspond to the menu-driven questions in the original version. These were used in a way that is similar to that of the LUKE system, i.e., the user is asked to assign grammaticality judgments to these sentences. Figure 6 shows the types of sentences that would be presented to the user. Note that the menu from the previous figure is given in parallel here to illustrate the correspondence between menu-driven questions and the relevant sentences. Once the

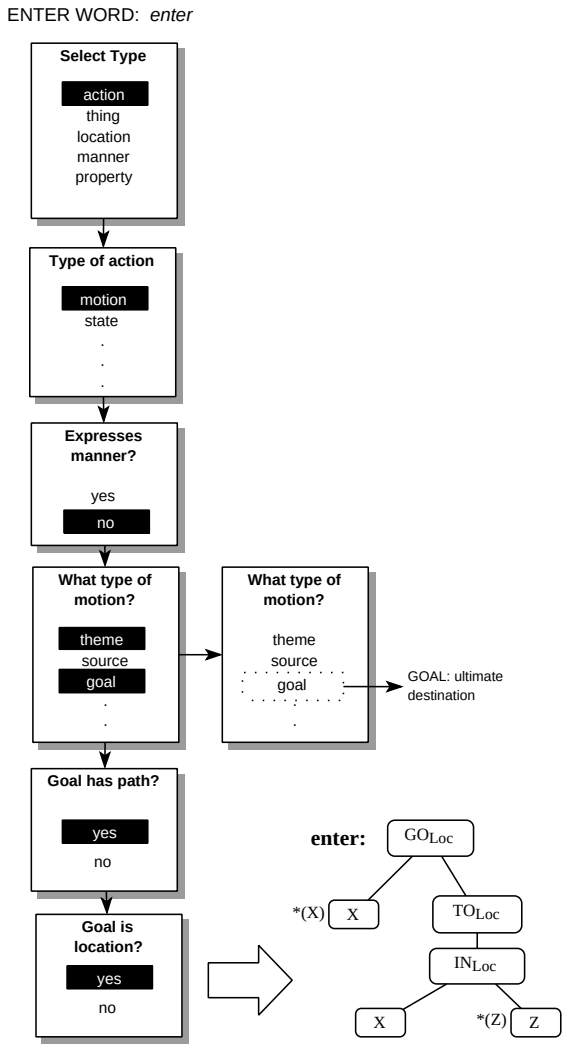

Figure 5. Another Semi-Automatic Tool: User's LCS Editor

user makes a judgment about these sentences, the information is then used to infer information about the LCS representation.

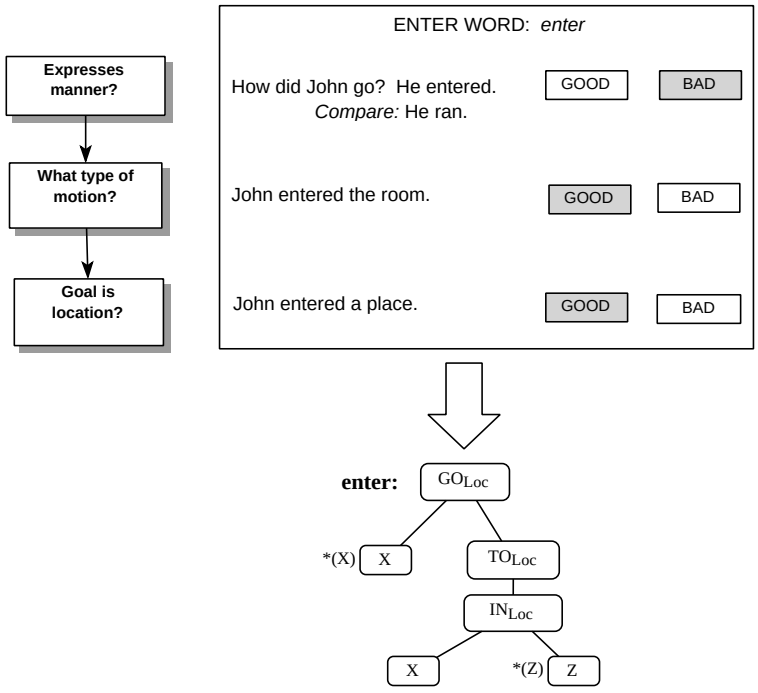

Figure 6. Enhanced User's LCS Editor

The key to the success of this enhanced design is the provision of an appropriate set of sentences that the user can easily verify and that provide enough information for automatic inference about the structure of the LCS. For example, the verb *enter* is different from *run* in that the former does not express manner of motion. This subtle distinction is brought out by asking the user to specify whether *How did John go?/He X-ed* is an appropriate sentence pair. For *run* this pair is appropriate since the "how" portion of the verb is inherent in its meaning—he went "runningly"; by contrast, for *enter* this pair is not appropriate since there is no "how" component of meaning.

If we were to push this design one step closer to automated lexical acquisition, we might consider building a sentence-finding program that searches a textual corpus (i.e., a large body of sentences in a particular language) for the occurrence of sentences (or sentence pairs) that correspond to the ones in the presentation window of Figure 6. Thus, instead of developing the appropriate set of sentences to isolate

meaning, the task is now to devise a program that is able to select appropriate sentences from corpora. We will discuss this possibility further in the next section.

5. Lexical Acquisition: Toward An Automated Approach

As machine-readable resources (i.e., online dictionaries, thesauri, and other knowledge sources) become more readily available to NLP researchers, automated acquisition has become increasingly more attractive. Several researchers have noted that the average time needed to construct a lexical entry can be as much as 30 minutes (see, e.g., (Neff and McCord, 1990; Copestake et al., 1995; Walker and Amsler, 1986)). Given that we are aiming for large-scale lexicons of 20-60,000 words, automation of the acquisition process is a necessity.

Previous research in automatic acquisition focuses primarily on the use of statistical techniques, such as bilingual alignment (Church and Hanks, 1990; Klavans and Tzoukermann, 1995; Wu and Xia, 1995), or extraction of syntactic constructions from online dictionaries and corpora (Brent, 1993; Dorr, Garman, and Weinberg, 1995; Yarowsky, 1995). Others have taken a more knowledge-based (interlingual) approach (Lonsdale, Mitamura, and Nyberg, 1995) to construction of lexical entries. The work of Copestake et al. (1995), like ours, falls between the purely syntactic and knowledge-intensive approaches. Both approaches attempt to use syntactic information (English-based grammatical codes) for acquisition of lexical representations; these representations are then ported to different languages by means of bilingual lexicons. Our approach is set apart from this earlier work in that it exploits certain linguistic constraints that govern the relation between syntactic structure and word meaning. We demonstrate that basic meaning components can be systematically derived from information about syntactic realizations; these meaning components are used to build LCSs which are then ported into different languages.

This approach extends the work of Levin (1993), applying her thesis (that syntax and semantics are systematically related) to the problem of lexicon construction.[2] The framework views verb behavior as a tool for "[probing] for linguistically relevant pertinent aspects of verb meaning" (Levin, 1993, p. 1). This basic idea has been applied to the problem of verb acquisition, where we are not "probing for" but indeed "acquiring" aspects of verb meaning from syntactic cues. Adopting this approach is not without experimental justification (Fisher, Gleitman, and Gleitman, 1991; Fisher et al., 1994; Gleitman, 1990; Landau and Gleitman, 1985). In particular, Fisher et al. (1994) present striking evidence that human acquisition of verb meanings relies overwhelmingly on syntactic information over other sources of information.

The determination of a word's meaning from its syntactic behavior is at the core of many other "bootstrapping" approaches as well. The work of Yarowsky (1995) demonstrates that it is possible to disambiguate word senses by assuming that words have one sense per "collocation" (nearby words). Our approach takes this a step further, however, showing that abstract structural relationships between a verb and its surrounding words (e.g., predicate-argument structure) can aid in determining

the verb's sense. Such information is frequently ignored in approaches that treat the input as an unparsed (but ordered) collection of words.[3]

An experimental corpus-based acquisition design is presented below. Next, we describe a verb classification technique that uses syntactic and semantic information extracted from alternative machine-readable resources. A primary objective of this technique is to demonstrate that synonymous verbs in an existing online resource, namely WordNet (Miller, 1986; Miller, 1990; Miller and Fellbaum, 1991) exhibit similar syntactic behaviors as characterized in the classification system of Levin (1993). We use this syntax-semantics relation to develop a lexical acquisition program that contributes both toward the enrichment of these online resources and toward the development of lexicons containing more complete information than is provided in either of these resources alone. Finally, a procedure that builds LCSs from classified verbs is described. These techniques have been used for construction of existing large-scale dictionaries for Arabic, Spanish, and Korean.

5.1. Corpus-Based Automatic Lexicon Construction

As described in the previous section, it is possible to derive the structure of the LCS representation automatically from a user's judgments about sentences such as those given in Figure 6. Figure 7 illustrates the proposed design of a corpus-based system in which the same types of sentences are analyzed, but without human interaction. In this example, the system is attempting to determine the LCS representation for the verb *run*. Each sentence that is found to be relevant in the corpus is analyzed for information that would add one component of meaning to the LCS representation. For example, the verb phrase *run quickly* in the first sentence indicates that this is a GO action; the addition of the manner component *runningly* can then be derived from the sentence pair, i.e., from the occurrence of a sentence that answers the question "how." Finally, the TO path is derived from the sentence *John ran into the room*, which indicates potential arrival at a destination.

The key to the success of this type of automatic design is to use an extensive corpus (for each of the languages of interest) and a "smart" analyzer. Unfortunately, as shown by Hogan and Levin (1994), it is a non-trivial task to acquire even simple components of meaning, such as argument structure information, through syntactic extraction from a corpus. Moreover, a corpus has inherent limitations; we have no guarantee that we will be able to find the precise sentences we need in order to predict LCS components of meaning. For example, even with a corpus as enormous as the Lancaster-Oslo-Bergen (LOB) corpus—60,000 sentences—there is no occurrence of the construction "How did `X Verb`", with or without a subsequent response of the form "`X Verb`ed."

The difficulty of searching for a small set of highly detailed constructions of this type has led us to investigate a higher level of abstraction that utilizes the verb classification presented in (Levin, 1993). Our goal is to test Levin's hypothesis that the syntactic behavior of a word is fully semantically determined and, moreover, that this property holds across all languages. For example, the verb *run* is a *Manner*

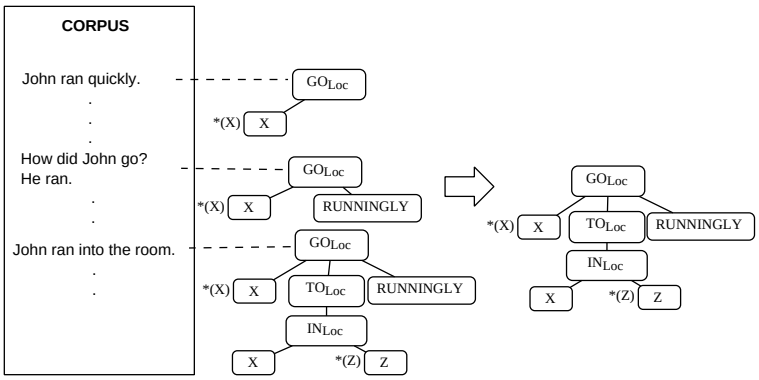

Figure 7. Toward Automatic Lexical Acquisition: Searching a corpus for sentences

of Motion verb which, according to Levin, participates in the following syntactic *alternations*, i.e., surface sentence pairs:

(10) (i) **Path:**
The horse ran
The horse ran through/into/out of the stream

(ii) **There Insertion:**
A horse ran out of the barn
There ran out of the barn a horse

(iii) **Induced Action:**
She ran the horse through the fields
The horse ran through the fields

(iv) **Locative Inversion:**
A horse ran out of the barn
Out of the barn ran a horse

(v) **Measure Phrase:**
We ran
We ran 5 miles

(vi) **Resultative Phrase:**
We ran
We ran ourselves into a state of exhaustion

Our technique involves searching a corpus for a wide range of alternation pairs of this type (e.g., "`X Verb`ed PP" coupled with "`Y Verb`ed `X` PP") rather than a small set of highly language-specific, interdependent pairs (e.g., "How did `X Verb`" immediately followed by "`X Verb`ed"). The idea is to determine the semantic class of a verb on the basis of its participation in one or both alternates of pairs such as those

given in (10). Suppose, for example, the acquisition program attempts to determine from a corpus the semantic class for the verb *run*; if the procedure finds a sentence that contains both *run* and a measure phrase, then it would postulate *Manner of Motion* as a possible semantic class.[4] If enough information is available to delineate a particular semantic class, the class can then be mapped into meaning components that are relevant to the LCS representation, e.g., the verb *run* as used above would be categorized such that its LCS would contain GO with a RUNNINGLY modifier and some sort of path.

Part of our investigation within this framework involved the compilation of a large matrix that crosses Levin's syntactic alternations (approximately 80, i.e., sections 1.1–8.6) with her semantic classes (approximately 192, i.e., sections 9.1–57), using Levin's publicly available online index (Levin, 1993) as a starting point. A portion of this matrix is given in Table 3. The entries in the table indicate, for each

Table 3. Deriving Semantic Classes from Syntactic Alternations

Syntactic Alternations	**Semantic Classes**					
	9.1	9.2	9.3	9.4	...	51.3.2
Path	N/Y	N/Y	Y/Y	Y/Y	...	Y/Y
There Insertion	Y/N	Y/N	Y/N	Y/N	...	Y/Y
Induced Action	Y/N	Y/N	Y/N	Y/N	...	Y/Y
Locative Inversion	N/N	N/N	N/N	N/N	...	Y/Y
Locative Preposition Drop	Y/N	Y/N	Y/N	Y/N	...	Y/Y
Measure Phrase	Y/N	Y/N	Y/N	Y/N	...	Y/Y
Resultative Phrase	Y/N	Y/N	Y/N	Y/N	...	Y/Y
Locative	Y/N	Y/N	Y/N	Y/N	...	Y/N
Middle	Y/N	Y/N	Y/N	Y/N	...	Y/N
Causative	Y/N	Y/Y	Y/N	Y/N	...	Y/N
Dative	N/N	N/N	N/N	Y/N	...	N/N
⋮	⋮	⋮	⋮	⋮	...	⋮

semantic class, which alternates in a pair are correlated with verbs in the class. For example, the semantic class 51.3.2 is a subclass of the *Manner of Motion* verbs; this class participates in both alternates of the Path pair given in (10) (indicated by Y/Y), but only one alternate of the Locative pair (indicated by Y/N), and neither alternate of the Dative pair (indicated by N/N).[5]

Preliminary results (Corbin, Copeland, and Buck, 1994) have demonstrated that this approach is easy to implement. However, there are still several inherent limitations to corpus-based searching for such constructions. For example, a pure syntactic parser is unable to distinguish among the following constructions:

(11) (i) **Fulfilling Construction:** John presented the student with an award.

(ii) **Locative Construction:** John loaded the truck with bricks.

(iii) **Instrument Construction:** John hit the wall with a hammer.

Each sentence has the basic syntactic frame `[np,v,np,pp(with)]`, yet these three constructions must necessarily be distinguishable in order to fully understand the meaning of the main verb.

An additional limitation of the approach concerns the issue of cross-linguistic applicability: it relies on a set of specific alternations that might have radically different analogs in languages other than English. Our own study of Korean has revealed that the syntactic tests for English do not apply directly to Korean verbs (Dorr et al., 1995a). An example is the Causative alternation, which is an important criterion for Levin's English verb classification. This alternation is irrelevant to Korean because it is not possible for a Korean verb to function as both a transitive and an intransitive verb. Rather, the passive alternation is used as an analog for such cases:

(12) (i) John-i changmwun-ul kkay-ss-ta [active]
John-Nom window-Acc break-Pst-Ind
'John broke the window'

(ii) changmwun-i kkay-ci-ess-ta [passive]
window-Nom break-Pass-Pst-Ind
'The window broke'
(The window was broken)

In this example, the Korean verb for *break* (root form 'kkay') is used in the active form for the transitive version and the passive form for the intransitive version. Similar studies have been conducted for other languages, e.g., for Japanese (Mitamura, 1990), French (Pugeault, Saint-Dizier, and Monteil, 1994; Saint-Dizier, 1996a; Saint-Dizier, 1996b), and Bangla, German, English, and Korean (Jones et al., 1994). Given that Levin's English classification is the result of a compilation of at least a decade of research articles and notes, it is clear that the construction of a table for every language is a non-trivial task.

It is important to note here that Levin does not assume that a verb in one language necessarily behaves the same as all of its equivalent forms in another language, especially if that verb has multiple verb senses. For example, there is no single Korean counterpart for all senses of the English verb *break*, so it should not be the case that *all* syntactic behaviors associated with the word *break* carry over to *all* possible counterparts in Korean. However, Levin (1993) argues that when the same alternations *do* exist across languages, the same semantic features are implicated. For example, the Australian language Warlpiri shows the conative alternation which, like English, involves motion/impact verbs (*She banged at the wall*). Other investigations have confirmed the existence of alternations in other languages, e.g., the Spray/Load construction (*John sprayed paint on the wall*, *John sprayed the wall with paint*) (Jones et al., 1994) and the middle construction (*The bread cuts easily*) (Kemmer, 1993).

Although many English alternations do not necessarily carry over directly to other languages, the semantic classes associated with these alternations are comprised of "basic meaning components" which are intended to be cross-linguistically

applicable. Thus, a verb in one language might not have an exact (identically classified) counterpart in another language, but the meaning components underlying two translationally related verbs (and other verbs in their corresponding semantic classes) are expected to be overlapping. The notion of "basic meaning components" is key here; if we decompose Levin's semantic classification into primitive units of meaning, then we are able to carry these meaning components over to other languages without necessarily requiring an intermediate step of constructing a language-specific semantic classification. (This is the approach that will be described in the next section.)

Other researchers have argued against the usefulness of Levin's semantic classification as the basis for defining verb classes cross-linguistically. An example is the work of Mitamura (1990) who shows a potential association between syntactic patterns and semantic patterns in Japanese, but then argues that the Japanese semantic classes are not the same as those developed by Levin for English. Mitamura claims that syntactic alternations in Japanese are based primarily on case distinctions; for example, verbs associated with the *ga* and *kara* case markers are in the semantic class of *giving* (send, report, give, etc.):

(13) (i) Watasi ga/kara Kimura-san ni tutaeta
'I reported to Mr. Kimura'

(ii) Kodomotati ga/kara sensei ni kaado o okutta
'The children sent a card to the teacher'

Since there is no semantic class of 'giving' of this type in (Levin, 1993), Mitamura argues that Levin's semantic classification is not useful for defining verbs cross-linguistically.

We claim that Mitamura's conclusion does not necessarily follow from her analysis. In fact, it is easy to find constructions that unify these three verbs in English, so it is not surprising that such constructions would exist in Japanese. Consider, for example, the construction `John X-ed the results to Mary`. All three verbs, *send* (Class 11.2), *report* (Class 37.7), and *give* (Class 13.1) fit grammatically into this construction, yet it is not the case that the Levin's system fails to adequately distinguish these three verbs semantically, nor is it the case that these three verbs have no meaning components in common. In an LCS-based system, these three verbs have the same basic LCS representation, which corresponds to a form of transfer:

(14) [CAUSE (X, [GO (Y, [TOWARD (Y, [AT (Y, Z)])])])]

The distinguishing component of meaning for *send*, *report*, and *give* resides elsewhere in the LCS, i.e., in the semantic field, as shown in Table 4.

As it turns out, this semantic distinction *does* have a syntactic reflex in English: the Dative construction distinguishes *send/give* from *report* and the From-To construction distinguishes *report/give* from *send*. Mitamura's example indicates that she has identified a very coarse-grained feature (the 'transfer' component of meaning—indeed that acquired by syntax in human experiments conducted by

Table 4. Distinguishing Among 'giving' Verbs

Verb	Levin Class	Field
Send	11.2 (Sending/Carrying verbs)	Loc
Report	37.7 (Say verbs)	Perc
Give	13.1 (Give verbs)	Poss

(Fisher, Gleitman, and Gleitman, 1991)), but perhaps has missed out on other semantic distinctions which may or may not have syntactic reflexes in Japanese. Other constructions (beyond *ga/kara*) would need to be examined in order to determine the full set of distinguishing features.

The approach described next relies heavily on the assumption that it is possible to map from Levin's semantic classes into the components of meaning relevant to the LCS representation. These meaning components, in turn, can be mapped into lexical items for different languages.

5.2. MRD-Based Automatic Lexicon Construction

As we saw above, there are a number of difficulties inherent in corpus-based acquisition—at least from the point of view of cross-linguistic applicability—due to the high degree of variation between corresponding syntactic constructions in different languages. However, we can still benefit from the syntax-semantics relationship if we can derive the underlying components of meaning from surface syntactic structures in a given language, and then carry these meaning components over to other languages. We adopt the view that it is the meaning components—not the syntactic structures associated with the meaning components—that capture cross-linguistic generalizations. Taking an approach that incorporates this view will allow us to avoid the tedium of specifying such a mapping on a per-language basis.

Our goal was to automatically classify 1400 "unknown" verbs, i.e., those not occurring in the Levin classification; these verbs were taken from English "glosses" (i.e., translations) provided in bilingual dictionaries for Spanish and Arabic.[6] Once the English verbs were classified, we applied a procedure called LEXICALL to build LCSs from the verbs in each class. (A sketch of this program is presented in the next section; additional details are provided in (Dorr, To appear in 1997).) Given that the LCS-based components of meaning are intended to carry over to other languages, this same LCS-building procedure was applied to verbs that were classified analogously in other languages.[7]

Unlike the corpus-based technique described in the previous section, our verb classification algorithm relies solely on machine-readable dictionaries (MRDs).[8] In particular, each unknown verb is assigned to a semantic class by examining the verb's synonyms from WordNet (Miller, 1986; Miller, 1990; Miller and Fellbaum, 1991)—an online lexical database containing thesaurus-like relations—and selecting those whose Levin class is associated with syntactic information matching that of the unknown verb. The syntactic information, which corresponds roughly to the

alternation pairs from (Levin, 1993), is represented by sets of codes adopted from the Longman's Dictionary of Contemporary English (LDOCE) (Procter, 1978) (e.g., D1 for Dative). Table 5 describes the significance of a subset of the syntactic codes in LDOCE. The total number of codes used by our algorithm is 174.

Table 5. Sample Syntactic Codes used in LDOCE

LDOCE Code	**Arguments**	**Adjuncts**	**Example**
D1	NP NP	—	She **allowed** him some money for expenses
D1-FOR	NP NP or NP PP[for]	—	She **built** him the house
D1-TO	NP NP or NP PP[to]	—	She **brought** him the newspaper
I	—	—	Olivier is **acting** tonight
I-AFTER	—	PP[after]	She **sought** after the truth
I-FOR	—	PP[for]	They **sought** for the right one
I-ON	—	PP[on]	He **acted** on our suggestion
I-UPON	—	PP[upon]	The drug **acted** upon the pain
L1	NP	—	He **acts** the experienced man
L9	ADV/PP	—	The play **acts** well
T1	NP	—	I **pioneered** the new land
T1-AT	NP	PP[at]	They **valued** the house at $300K
T1-INTO	NP	PP[into]	We **initiated** him into the group
T1-FOR	NP	PP[for]	They **admired** him for his stamina
T1-OF	NP	PP[of]	He **cleared** the table of dishes
T1-TO	NP	PP[to]	She **escorted** him to the prom
T3	VP[to+inf]	—	He **tried** to do it
T4	VP[+prog]	—	She **tried** eating the new food
T5	CP[that+fin]	—	I **know** that he is happy
WV4	-ing adjectival	—	I've had a **trying** day
WV5	-ed adjectival	—	He was convicted for **attempted** murder
N (denominal verb)	—	—	**pioneer** (noun)

The intent of the algorithm is to: (1) classify verbs that were not previously defined in Levin's book using the verbs WordNet synonyms and LDOCE codes as a starting point; (2) demonstrate that synonymous verb senses share distributional patterns. The basic idea is to first determine the most likely candidates for semantic classification of a verb by examining the verb's synonym sets, many of which intersect directly with the verbs classified by Levin. The "closest" synonyms are then selected from these sets by comparing the LDOCE codes of the unknown word with those associated with the semantic class of each synonym.

An investigation into a possible correlation between LDOCE codes and the syntactic alternations of Levin was previously undertaken by Boguraev and Briscoe (1989), but only for the dative alternation. Their study involved an analysis of human judgments of meaning for those verbs containing dative-like codes (e.g., the D1 code mentioned above). The results of this study indicated that, on the basis of human judgments, membership in the semantic class of "Change of Possession" does not correlate well with the occurrence of dative-like codes. However, as the authors themselves indicate (p. 113), the dative alternation is only one, among many alternations that collectively make up the semantic classification of Levin. Even more importantly, the dativizing verbs are not solely restricted to the "Change

of Possession" verb class in (Levin, 1993); several other classes contain dativizing verbs, including "Having Verbs", "Send Verbs", "Slide Verbs", "Carry Verbs", "Drive Verbs", "Throw Verbs", "Transfer of a Message Verbs", etc.[9] We claim that a larger scale investigation of the code-class correlations based on more specialized code pairs, e.g., D1/D1-TO/D1-FOR/D1-AT rather than just D1 (see Table 5) is a more promising approach, especially now that the semantic classification of Levin (1993) significantly refines that used in this earlier study. In addition, we take the view that it is necessary to examine *all* alternations (those both legal and illegal with verbs) before determining class membership, given the potentially wide range of combinations and interactions.

Others who have used the LDOCE for automatic extraction (e.g., Alshawi (1989); Wilks et al. (1989)) have focused either on the prediction of syntactic phrase structure or on generation of semantic relations from definition analyses rather than on the derivation of an interlingual representation from grammar code combinations. The use of bilingual dictionaries for the automatic construction of multi-language information from LDOCE has also been investigated by Farwell, Guthrie, and Wilks (1993). This work is closely related to ours in that it focuses on the extraction of broad semantic restrictions on arguments.[10] The work of Sanfilippo and Poznanski (1992, p. 82) is even more closely related to our approach in that they attempt to create a large lexical database using semi-automatic techniques to recover syntactic and semantic information from machine-readable dictionaries. However, they claim that the semantic classification of verbs based on standard machine-readable dictionaries (e.g., the LDOCE) is "a hopeless pursuit [since] standard dictionaries are simply not equipped to offer this kind of information with consistency and exhaustiveness."

Our work challenges claims about the infeasibility of automatic lexical acquisition from MRDs. We claim that a strong correlation exists between certain LDOCE code combinations and the semantic classes associated with WordNet synonyms. In particular, we show that synonymous verbs in WordNet exhibit similar syntactic behaviors, as characterized in the classification system of (1993), except for a small residue of cases (17% in our sample).

As a starting point, we assigned canonical and prohibited LDOCE codes to each semantic class by automatically extracting basic syntactic patterns from all of the sentences in Levin's book (Dorr and Jones, 1996a; Dorr and Jones, 1996b). Legal and illegal syntactic patterns were prefixed as "1-" and "0-", respectively.[11] These patterns were mapped into LDOCE codes and grouped into canonical and prohibited codes for each class. Table 6 and 7 show the code groupings for a subset of Levin's classes (numbers between 9.1 and 57) and new classes (numbers between 001 and 026); the hand-mappings between syntactic patterns and LDOCE codes are included for completeness. Note that some of the patterns (e.g., the Middle Construction *The bread cuts easily*) do not map into LDOCE codes. Table 8 shows the full set of canonical and prohibited LDOCE codes for each of these classes.

Next, we developed a verb classification algorithm that operates in the following way. If a given verb V is in Levin, it is classified directly. Otherwise, V's WordNet

Table 6. Mapping between Syntactic Patterns and LDOCE Codes

Class	Syntactic Pattern	LDOCE Codes
10.1 Remove	1-[np,v,np,pp([from,around])]	T1-FROM X9
	1-[np,v,np,pp([from,under])]	T1-FROM X9
	1-[np,v,np,pp(from)]	D1-FROM T1-FROM
	0-[np,v,np,pp([off,of])]	T1-OFF X9 X9-OFF
	0-[np,v,np,pp([out,of])]	D1-OUT_OF T1-OUT_OF X9-OUT_OF
	0-[np,v,np,pp(of)]	D1-OF T1-OF
	0-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
	0-[np,v,pp(at),pp(from)]	I-AT I3-AT L9-AT WV4-AT
	0-[np,v,pp(from)]	I-FROM
10.2 Banish	1-[np,v,np,pp(from)]	D1-FROM T1-FROM
	1-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
	0-[np,v,np,pp(of)]	D1-OF T1-OF
	0-[np,v,pp(at),pp(from)]	I-AT I3-AT L9-AT WV4-AT
	0-[np,v,pp(from),pp(to)]	I-FROM I-TO
	0-[np,v,pp(from)]	I-FROM
10.5 Steal	1-[np,v,np,pp(for)]	T1-FOR D1-FOR
	1-[np,v,np,pp(from)]	D1-FROM T1-FROM
	0-[np,v,np,np]	D1 X1
	0-[np,v,np,pp(of)]	D1-OF T1-OF
	0-[np,v,pp(at),pp(from)]	I-AT I3-AT L9-AT WV4-AT
	0-[np,v,pp(from)]	I-FROM
13.4.1 Fulfilling	1-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
	1-[np,v,np,pp(with)]	D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH
	0-[np,v,np,pp([next,to])]	X9
	0-[np,v,np,pp(at)]	D1-AT T1-AT X9-AT
	0-[np,v,np,pp(near)]	X9
	0-[np,v,np,pp(onto)]	D1-ONTO T1-ONTO X9
22.2 Amalgamate	1-[np(and),v]	I-FROM I-WITH
	1-[np,v,adv(easily)]	—
	1-[np,v,np(and)]	T1 I-FROM I-WITH
	1-[np,v,np,pp(with)]	D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH
	1-[np,v,np]	T1 L1
	1-[np,v,pp(with)]	I-WITH I3-WITH L9-WITH T5-WITH T6-WITH
	1-[np,v]	I
	0-[np(and),v,adv(together)]	I-TOGETHER I3-TOGETHER WV3-TOGETHER L9-TOGETHER
	0-[np,v,np(and),adv(together)]	T1-TOGETHER X9-TOGETHER
29.4 Declare	1-[np,v,np,infinitive]	D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE
	1-[np,v,np,np]	D1 X1
	1-[np,v,s_comp]	T5 I5 X5
	0-[np,v,np,pp(as)]	T1-AS X9-AS X1-AS
	0-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
29.5 Conjecture	1-[np,v,np,infinitive]	D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE
	1-[np,v,s_comp]	T5 I5 X5
	0-[np,v,np,np]	D1 X1
	0-[np,v,np,pp(as)]	T1-AS X9-AS X1-AS

Table 7. Mapping between Syntactic Patterns and LDOCE Codes (Continued)

Class	Syntactic Pattern	LDOCE Codes
30.2 Sight	1-[np,v,np,vp(ing)]	V4 V4-FROM V4-WITH X4
	1-[np,v,np]	T1 L1
	0-[np,v,adv(easily)]	—
	0-[np,v,np,vp(bare)]	V2 T2
	0-[np,v,np]	T1 L1
	0-[np,v,pp(at)]	I-AT I3-AT L9-AT WV4-AT
	0-[np,v,s_comp]	T5 I5 X5
31.1 Amuse	1-[adjective(ing)]	WV3 WV4
	1-[expl(it),v,np,infinitive]	D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE
	1-[expl(it),v,np,s_comp]	D5
	1-[infinitive,v,np]	T1 I3
	1-[np,aux(be),v,pp(at)]	WV5 WA5
	1-[np,aux(be),v,pp(by)]	WV5 WA5
	1-[np,aux(be),v,pp(with)]	WV5 WA5
	1-[np,v,adv(easily)]	—
	1-[np,v,np,adjective]	X7
	1-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
	1-[np,v,np,pp(with)]	D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH
	1-[np,v,np]	T1 L1
	1-[np,v,pp(to)]	I-TO
	1-[s_comp,v,np]	T1
	0-[np,v,pp(at)]	I-AT I3-AT L9-AT WV4-AT
	0-[np,v]	I
33 Judgment	1-[np,v,np,pp(as)]	T1-AS X9-AS X1-AS
	1-[np,v,np,pp(for)]	T1-FOR D1-FOR
	1-[np,v,np]	T1 L1
	0-[np,v,adv(easily)]	—
	0-[np,v,np,np]	D1 X1
	0-[np,v,np,pp(in)]	D1-IN T1-IN
37.7 Say	1-[np,v,np,pp(to)]	D1-TO T1-TO WV5-TO
	1-[np,v,pp(to),s_comp]	D5-TO T5-TO
	1-[np,v,s_comp]	T5 I5 X5
	0-[np,v,np,np]	D1 X1
	0-[np,v,pp(about)]	I-ABOUT
	0-[np,v,pp(to)]	I-TO
54.4 Price	1-[np,v,np,pp(at)]	D1-AT T1-AT X9-AT
	1-[np,v,np]	T1 L1
	0-[np,v,np]	T1 L1
	0-[np,v,pp(at)]	I-AT I3-AT L9-AT WV4-AT
002 Coerce	1-[np,v,np,infinitive]	V3
	1-[np,v,np,pp([into])]	T1-INTO
005 Aspire	1-[np,v,infinitive]	T3
017 Claim	1-[np,v,infinitive]	T3
	1-[np,v,s_comp]	T5 T5A T5B T5C

Table 8. Canonical and Prohibited LDOCE Codes for Semantic Classes

Class	Canonical LDOCE Codes	Prohibited LDOCE Codes
10.1	D1-FROM T1-FROM X9	I-FROM D1-OF T1-OF D1-OUT_OF T1-OUT_OF X9-OUT_OF D1-TO T1-TO WV5-TO I-AT I3-AT L9-AT WV4-AT T1-OFF X9-OFF
10.2	D1-FROM T1-FROM D1-TO T1-TO WV5-TO	I-TO I-FROM D1-OF T1-OF I-AT I3-AT L9-AT WV4-AT
10.5	D1-FROM T1-FROM T1-FOR D1-FOR	I-FROM D1 X1 D1-OF T1-OF I-AT I3-AT L9-AT WV4-AT
13.4.1	D1-TO T1-TO WV5-TO D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH	D1-AT T1-AT X9-AT D1-ONTO T1-ONTO X9
22.2	I D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH I-FROM I-WITH I3-WITH L9-WITH T5-WITH T6-WITH T1 L1 I-TO	I-TOGETHER I3-TOGETHER WV3-TOGETHER L9-TOGETHER T1-TOGETHER X9-TOGETHER
29.4	D1 X1 D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5	D1-TO T1-TO WV5-TO T1-AS X9-AS X1-AS
29.5	D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5	D1 X1 T1-AS X9-AS X1-AS
30.2	T1 L1 V4 V4-FROM V4-WITH X4	I-AT I3-AT L9-AT WV4-AT T5 I5 X5 V2 T2
31.1	D5 I-TO X7 D1-TO T1-TO WV5-TO D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE I3 T1 L1 WV3 WV4 WV5 WA5	I I-AT I3-AT L9-AT WV4-AT
33	T1 L1 T1-AS X9-AS X1-AS T1-FOR D1-FOR	D1 X1 D1-IN T1-IN
37.7	D1-TO T1-TO WV5-TO D5-TO T5-TO T5 I5 X5	I-ABOUT I-TO D1 X1
54.4	D1-AT T1-AT X9-AT T1 L1	I-AT I3-AT L9-AT WV4-AT
002	V3 T1-INTO	—
005	T3	—
017	T3 T5 T5A T5B T5C	—

synonyms are extracted. If none of V's WordNet synonyms are in Levin, V is set aside for a later application of the algorithm (e.g., after one or more of its synonyms have been classified). Otherwise, V is classified in one of two possible ways: (1) If V's LDOCE codes do not match the canonical LDOCE codes for any semantic class associated with the WordNet synonyms of V, a new class is created;[12] (2) If V's LDOCE codes match the canonical LDOCE codes for a semantic class associated with the V's synonyms, V is classified in that class. The notion of "match" is based on the degree to which there is an intersection between V's LDOCE codes and the canonical LDOCE codes for a candidate class, with a preference for those classes whose prohibited LDOCE codes are not among V's LDOCE codes.

1. If V is a verb in Levin, then return V's semantic class(es) directly.
2. Find V's synonym set S from WordNet.
3. If V has no WordNet synonym set, then put V aside (for later classification) and return nil.
4. Produce a candidate set C of semantic classes (from Levin) corresponding to the elements of S.
5. If C is empty or if the canonical LDOCE codes associated with each $C_i \in C$ are completely mismatched with V's LDOCE codes, then hypothesize a new class and return this class.
6. Return all classes $C_k \in C$ whose canonical LDOCE codes most closely match the LDOCE codes for V, using a weighting heuristic to resolve ties.[1]

Figure 8. Semantic Classification Algorithm for New Verbs

More formally, the classification algorithm for a verb V consists of the steps shown in Figure 8. This procedure is run iteratively, i.e., after 100-200 verbs are classified, the procedure is re-run on the remaining set of unknown verbs with the larger database of semantic classes. In our initial experiment, we focused on 95 verbs not in (Levin, 1993) taken from the LDOCE *control vocabulary*, i.e., primitive words used for defining dictionary entries. These 95 verbs are shown in Figure 9.[14]

A total of 134 semantic class assignments were made by the algorithm, with several verbs receiving more than one class assignment. Of these, 82 (61% of the total number of assignments) were hand-verified to be correct. The full set of correct assignments is shown in Table 9 (50 assignments to existing Levin classes) and Table 10 (32 assignments to new classes). In addition to the word and its assigned class, these tables show the LDOCE codes associated with the verbs, and the set of matching synonyms from that class. The LDOCE codes relevant to the

afford, aim, answer, attack, attempt, attend, attract, bar, bear, become, beg, belong, blame, branch, breed, broadcast, bury, calculate, command, consist, curse, damage, dare, deal, deceive, decide, defeat, defend, demand, deserve, do, doubt, dream, educate, equal, expect, experience, fail, farm, force, forget, fulfil, govern, help, include, influence, inform, inquire, insure, interrupt, lack, let, limit, manage, match, mend, mistake, park, participate, pause, permit, persuade, plan, possess, practise, preserve, pretend, prevent, protect, reduce, refuse, repair, reply, request, retreat, safeguard, seem, shade, share, spell, spend, spoil, spring, step, succeed, suit, swear, tend, test, tidy, tour, translate, understand, undo, urge

Figure 9. 95 Control Verbs from LDOCE not found in (Levin, 1993)

class assignment (those that intersect the canonical LDOCE codes for the assigned class) are underlined.

In Table 9, codes in brackets [ ] are relevant to the verb, but do not occur in the verb's LDOCE entry. These codes correspond to legal sentences that were hand-constructed during human verification of the output. A bracketed code indicates that the associated verb would have been *uniquely* assigned to the correct class(es), had the code been included in the verb's LDOCE entry. In actuality, without the bracketed code, the algorithm has assigned the verb to the correct class (as shown in the table), but also to other incorrect class(es) given in a different table below.[15] For example, the verb *curse* has been assigned correctly to class 33, "Judgment Verbs", but it has also been assigned incorrectly to class 10.1, "Remove Verbs." One of the canonical codes that distinguishes class 33 from class 10.1 is T1-FOR (see Table 8). This code does not occur in the LDOCE entry for *curse*, yet it is clearly relevant to the verb, e.g., *She cursed him for his arrogance*. It is therefore erroneously absent from the LDOCE. Such omissions from MRDs might lead to overgeneralizations in lexical acquisition that produce both correct and incorrect class assignments. Fortunately, the number of incorrect assignments based on overgeneralizations is low enough (22% of the total number of assignments) to benefit from the overall approach. We return to such cases shortly.

The identification of new semantic classes (Table 10) is a side benefit of our experiment. The acquisition algorithm hypothesized a new class each time it encountered a verb whose LDOCE codes did not match the canonical codes of the synonyms' classes. For example, the LDOCE codes associated with the verb *force* did not match the canonical codes of any of the classes associated with its synonym *push*. The program detected that the mismatched code T1-INTO (as in *She forced him into making the decision*) was associated with *force* and *push* as well as several other synonyms: *allure*, *badger*, *bamboozle*, *blackmail*, *bribe*, *browbeat*, *bulldoze*, *bully*, *cajole*, *coax*, *coerce*, *con*, *delude*, *dupe*, *ensnare*, *entrap*, *lead*, *mislead*, *persuade*, *pressure*, *push*, *railroad*, *rush*, *shame*, *shanghai*, etc. Levin's framework

Table 9. Correct Assignments to Levin's Semantic Classes (50)

Word	Assigned Class	LDOCE Codes	WordNet Synonyms
afford	13.1	D1 T1 T1-TO T3	render give
aim	40.3.1	I I-AT I-FOR I3 T1 T1-AT N	point
answer	37.1	I I-FOR I-TO I-WITH T1 N	tell
attack	33	I T1 N [T1-FOR]	criticize
blame	33	D1-FOR D1-ON T1 T1-FOR N	impeach fault criticize
branch	23.4	I N [I-FROM]	diverge
breed	22.2	I T1 N [T1-WITH]	mate pair
bury	9.1	T1 X9	situate immerse
bury	16	T1 X9	conceal hide
command	37.2	I T1 T5B T5C V3 N	tell
curse	33	I T1 N [T1-FOR]	abuse
damage	31.1	T1 WV4 N	devastate afflict
deal	13.1	D1 D1-TO I T1 N	give
deal	13.3	D1 D1-TO I T1 N	allot
deceive	33	I T1 T1-IN T1-INTO WV4	victimize
doubt	29.5	T1 T5A T6A N	suspect
doubt	31.2	T1 T5A T6A N	distrust
dream	31.3	I I-ABOUT I-OF T1 T5A N	feel
educate	29.2	T1 [V3]	train
farm	26.5	I T1 N	collect
fulfil	13.4.1	T1	supply provide
help	13.4.1	I T1 T1-TO T4 V2 V3 V4 N [T1-WITH]	supply provide serve
influence	31.1	T1 V3 N	tempt charm touch affect
inquire	31.3	I I-ABOUT T1 T6A	wonder
interrupt	55.1	I T1 [L9]	terminate end
lack	32.1	T1 N	need want
manage	30.2	I T1 T3	watch
mend	45.4	I T1 N	improve heal ameliorate
park	9.1	I T1 X9 N	position place set put
pause	53.1	I N	delay hesitate
permit	29.5	I T1 T4 V3 X9 N	allow
possess	47.8	T1 WV6	dominate
practise	14	I I-AS T1 T1-ON T1-UPON T4 WV4	read study learn
preserve	13.5.1	T1 T1-FROM WV5 N	keep save
reduce	26.6	I T1 T1-FROM T1-TO	turn change
repair	45.4	I T1 N	improve ameliorate
reply	37.7	I I-TO T5 N	say state
safeguard	29.8	T1 N	guard escort
spell	37.1	D1 I T1 T1-WITH N	recite write
spoil	31.1	I T1 T1-OF WV5 N	devastate afflict discourage frustrate baffle
spring	51.3.1	I L7 L9 T1 T1-ON X7 N	bounce move
step	11.2	I L9 T1 N	move
step	51.3.1	I L9 T1 N	move
step	51.3.2	I L9 T1 N	travel
succeed	47.8	I I-IN I-TO T1	follow
swear	37.7	I I-AT T1 T1-ON T1-TO T3 T5	declare
tidy	45.4	I T1 N ADJ	straighten neaten
tour	51.3.2	L9 N	journey travel
translate	26.6	I T1	alter change transform
undo	44	T1	wreck ruin destroy

Table 10. Correct Assignments to New Semantic Classes (32)

Word	Assigned Class	LDOCE Codes	WordNet Synonyms
attempt	005	T1 T3 T4 WV5 N	try seek
attend	013	I I-ON I-TO I-UPON T1	serve
bear	014	D1 D1-FOR D1-TO I L9 T1 T4 X9 N	take make pay
beg	015	I I-FOR I3 T1 T1-OF T3 T5C V3	pray ask
consist	022	T1 T4 WV6 [L9]	lie
decide	017	I T1 T3 T5A T6A T6B V3	{make get}
defend	021	T1 T1-AGAINST T1-FROM T4	guard keep
demand	017	T1 T3 T5C N	claim
do	006	D1 D1-FOR I I-FOR I2 L7 L9 T1 WV2 N	{pass}
expect	015	T1 T1-FROM T3 T5A T5B V3 WV6 X9	wish hope desire
fail	005	I I3 T1 T3 N	omit
force	002	T1 T1-INTO V3 WV5 X7 N	push
forget	005	I T1 T3 T4 T5A T6A T6B	omit
include	024	T1 T4 [X9]	admit allow
insure	021	T1 T1-AGAINST T5 [T1-FROM]	guard
let	019	T1 T1-FROM T1-TO V2 N	{rent}
manage	005	I T1 T3	watch
permit	024	I T1 T4 V3 X9 N	allow
persuade	002	D5 T1 T1-INTO T1-OF T1-OUT_OF V3	{make}
plan	005	I I-FOR I-ON T1 T3 N	think intend mean propose
prevent	023	T1 V4 [T1-FROM]	keep
protect	021	T1 T1-AGAINST T1-FROM	guard
refuse	005	D1 I T1 T3 N	decline
request	015	T1 T5 V3 N [T3]	ask
request	017	T1 T5 V3 N [T3]	ask
seem	020	I-IT L1 L1-TO_BE L7 L7-TO_BE L9 L9-TO_BE WV6	appear
shade	021	L9 T1 N [T1-AGAINST] [T1-FROM]	{draw}
share	026	I T1 T1-AMONG T1-BETWEEN T1-WITH N	distribute
spend	025	I T1 T1-FOR T1-IN T1-ON	pay
tend	005	L9 T1 T3	think watch
understand	018	I T1 T1-BY T5 T6 V3 WV6	feel perceive understand
urge	002	T1 T1-ON V3 X9 N [T1-INTO]	press

does not contain a semantic class that unifies these verbs, yet, coincident with our hypothesis, all of these verbs are semantically related.

In Table 10, bracketed codes [ ] are similar in nature to those listed in Table 9, i.e., these are codes that do not occur in the verb's LDOCE entry. Unlike the entries in the first table, however, none of the entries in the second table were assigned to other (incorrect) classes. Bracketed codes correspond to LDOCE codes that were required for membership in new classes; these codes were identified as such by the program after other verbs were assigned to the new class. For example, the code T5 occurs in the LDOCE entry for *request* (e.g., *He requested that she bring the book*). This is a canonical code for the new class 017 "Claim Verbs." However, an additional code, T3 (e.g., *He requested to go home*) was identified by the program as a canonical code for class 017 once other members associated with that code (e.g., *decide* and *demand*) were encountered (see Table 8). Thus, since *request* was assigned to class 017, it inherited this new code, even though the code does not occur in the LDOCE entry for *request*. This example illustrates how our lexical acquisition techniques are useful, not only for verb classification, but for enhancement of LDOCE as an online resource.

Table 10 includes one additional notation, i.e., verbs enclosed in braces { }, which refer to WordNet synonyms that were automatically ruled out by the program as members of the newly identified class. Because the mismatched codes (which make up the canonical code set of the new class) are not automatically associated with the verb's synonyms, it is often the case that the verb's synonyms will not necessarily be assigned to that new class. For example, a WordNet synonym of *persuade* is *make*, but *make* does not fit appropriately into the new class 002 "Coerce Verbs" because its intersecting LDOCE codes do not intersect the canonical codes (V3, T1-INTO) for the new class (see Table 8).

In our experiment, we arrived at 26 new semantic classes based on our algorithm. Table 11 shows the classes with some example verbs. The labels for these classes were assigned by hand. Some of the verbs were classified in later iterations of the acquisition algorithm, after the initial control vocabulary was classified.

In addition to correct assignments to (existing or new) semantic classes, the algorithm produced 29 incorrect assignments (22% of the total number of assignments) that were later hand-verified to be the result of syntactic omissions in either LDOCE or Levin (see Table 12). In such cases, the relevant WordNet synonym was available, but the canonical/prohibited codes associated with the synonym's class(es) were not specific enough for the class(es) to be selected. One type of syntactic omission (the vast majority of such cases) arises from missing syntactic codes in LDOCE. For example, T1-FROM is not included in the LDOCE entry for *bar*, yet this code is relevant to *bar*, as in *They barred him from office*). Such omissions are labeled 'LD' in Table 12.

A second type of syntactic omission arises when a relevant LDOCE code is not associated with a semantic class because there is no sentence from Levin's data corresponding to this code. For example, there is no sentence corresponding to T1-

Table 11. New Semantic Classes Identified During Acquisition

Class		LDOCE Codes	New Verbs
001	Be-For Verbs	I-FOR	be suited for, be proper for, ...
002	Coerce Verbs	V3 T1-INTO	allure, bribe, cajole, coerce, persuade, urge, ...
003	Conspire Verbs	T1-AGAINST	legislate, protest, rebel, ...
004	Drop Verbs	L9	let drop, let fall, let flow, ...
005	Aspire Verbs	T3	aspire, attempt, deserve, forget, manage, plan, refuse, tend, try, ...
006	Do Verbs	L7 L9 D1 D1-FOR	do, re-do, ...
007	Exceed Verbs	I-IN	exceed, excel, outbid, outnumber, succeed, transcend, ...
008	Impose Verbs	T1-ON T1-UPON	hazard, enforce, exert, gamble, impose, ...
009	Play Verbs	I-WITH I-AGAINST	hock, pawn, play with, ...
010	Become Verbs	L1 L7	become bored with, become impatient with, ...
011	Penetrate Verbs	I-THROUGH WV5-WITH	penetrate, permeate, pervade, ...
012	Sustain Verbs	WV4	keep to, maintain, stick to, sustain, ...
013	Attend Verbs	I-ON	attend, serve, ...
014	Bear Verbs	D1 D1-FOR D1-TO X9	bear, take
015	Beg Verbs	T3 V3	ask, beg, dare, expect, need, pray, request, wish
016	Attract Verbs	L9 WV5 X9	attract, ...
017	Claim Verbs	T3 T5 T5A T5B T5C	ask, claim, decide, demand, pretend, request
018	Perceive Verbs	T5 T5A T5B T5C T6 T6A T6B V3	feel, perceive, understand
019	Let Verbs	V2	let, make
020	Seem Verbs	L1-TO_BE	appear, seem
021	Defend Verbs	T1-AGAINST T1-FROM	defend, guard, insure, protect, shade
022	Consist Verbs	L9 WV6	belong, consist ...
023	Prevent Verbs	V4 T1-FROM	keep, prevent
024	Admit Verbs	T4 X9	admit, permit, include, allow
025	Spend Verbs	T1-ON T1-FOR	spend, pay
026	Share Verbs	T1-AMONG	distribute, share

AGAINST in Levin's semantic class 22.2, yet *pair* permits this code, as in *They were paired against more powerful forces*. Such omissions are labeled 'LE_1' in Table 12.

A third type of syntactic omission arises when a WordNet synonym is in Levin, but not in a class relevant to the verb under consideration. For example, in Table 12 the verb *retreat* is considered synonymous with *withdraw*, but only in the sense provided by class 10.2 (Banish Verbs), not in the senses provided by the other classes in which *withdraw* occurs (10.1 Remove and 10.5 Steal). In such cases the program discovered a discrepancy in the canonical LDOCE codes and predicted a new class; however, our hand verification revealed that *withdraw* belongs in a third class and, if it had been there, *retreat* would have been classified correctly. Such omissions are labeled 'LE_2' in Table 12.

Table 12. Incorrect Classifications Resulting from Syntactic Omissions

Verb	**Assigned Class**	**Corrected Class**	**WordNet Synonym**	**Syntactic Omissions (LD)OCE or (LE)vin**
attack	29.6	33	criticize	LD: Add T1-FOR to 'attack'
attack	36.1	33	criticize	LD: Add T1-FOR to 'attack'
attack	36.3	33	criticize	LD: Add T1-FOR to 'attack'
bar	16	10.2	expel	LD: Add T1-FROM to 'bar'
branch	10.1	23.4	diverge	LD: Add I-FROM to 'branch'
branch	23.1	23.4	diverge	LD: Add I-FROM to 'branch'
branch	9.11	23.4	diverge	LD: Add I-FROM to 'branch'
breed	27	22.2	mate	LD: Add T1-WITH to 'breed'
breed	36.1	22.2	mate	LD: Add T1-WITH to 'breed'
breed	45.4	22.2	mate	LD: Add T1-WITH to 'breed'
calculate	29.1	54.4	estimate	LD: Add T1-AT to 'calculate'
curse	10.1	33	abuse	LD: Add T1-FOR to 'curse'
defeat	22.3	42.1	kill	LD: Add D1-WITH to 'defeat'
defeat	26.5	42.1	kill	LD: Add D1-WITH to 'defeat'
educate	45.4	29.2	train	LD: Add V3 to 'educate'
experience	29.5	30.1	feel	LD: Add V4 to 'experience' and 'feel'
inquire	36.1	31.3	communicate	LD: Add T6 to 'inquire'
interrupt	23.2	55.1	terminate	LD: Add L9 to 'interrupt'
interrupt	40.8.3	55.1	terminate	LD: Add L9 to 'interrupt'
interrupt	45.1	55.1	terminate	LD: Add L9 to 'interrupt'
interrupt	48.1.1	55.1	terminate	LD: Add L9 to 'interrupt'
limit	22.4	47.8	contain	LD: Add T1-TO to 'contain'
match	9.8	22.2	pair	LE_1: Add T1-AGAINST to 22.2
mistake	31.1	29.2	identify	LD: Add X1-AS,X7-TO_BE to 'mistake' and 'identify'
retreat	NEW	10.2	withdraw	LE_2: Add 'withdraw' to 10.2
spoil	31.1	45.5	decay	LD: Add WV5 to 'decay'
succeed	51.6	47.8	follow	LD: Add L1 to 'succeed'
suit	45.4	54.3	fit	LD: Add T1-IN to 'suit'
test	31.1	35.4	examine	LD: Add T1-ON to 'test'

An additional 23 incorrect assignments (17%) were later hand-verified to be the result of a complete semantic mismatch between WordNet and Levin. These are shown in Table 13 below. As an example, consider the verb *govern*. This has the WordNet synonym *order*, one sense of which Levin classifies into class 13.5.1 (Get

Verbs). This is not the sense of *order* that corresponds to the verb *govern*. In this case, the hand-corrected class is 30.2 (Sight Verbs), including *inspect*, *survey*, etc. In some cases, the hand-corrected class does not occur in Levin.[16] An example is the verb *deserve*, which is assigned to the new class 005 (Aspire Verbs).

Table 13. Incorrect Classifications Resulting from WordNet/Levin Semantic Mismatches

Verb	**Assigned Class**	**Corrected Class**	**Mismatched Synonym(s)**	**LDOCE Code(s) for Hand-Corrected Class**
attract	12	016	draw pull	L9 WV5 X9
attract	23.2	016	draw pull	L9 WV5 X9
become	51.1	010	fall come go	L1 L7
belong	47.6	022	lie	L9
belong	50	022	lie	L9
broadcast	NEW	37.4	bare spread beam distribute	I-TO T1-ABOUT
dare	29.6	015	act	T3 V3
deserve	29.1	005	rate	T3
deserve	29.6	005	rate	T3
deserve	54.4	005	rate	T3
equal	45.4	54.2	regularize	L1
govern	30.2	13.5.1	order	T1
govern	30.2	29.3	rule	T1
govern	30.2	45.4	regularize	T1
inform	36.1	37.9	communicate	T1-ABOUT
participate	29.2	36.1	enter	I-WITH
participate	29.6	36.1	act	I-WITH
participate	51.1	36.1	enter	I-WITH
pretend	29.2	017	represent	T3 T5
pretend	29.6	017	act	T3 T5
pretend	36.3	017	play	T3 T5
succeed	51.6	007	follow	I-IN
succeed	47.8	007	follow	I-IN

We now consider three examples that demonstrate how algorithm works in more detail.

Example 1: Semantic classification of *swear*. The LDOCE specification for this verb is I I-AT T1 T1-ON T1-TO T3 T5. Using the synonymy feature of WordNet, Step 4 of the classification algorithm automatically extracts four candidate classes associated with the synonyms of this word: (1) Class 29.4 "Declare Verbs" (*declare, avow*); (2) Class 29.5 "Conjecture Verbs" (*assert*); (3) Class 37.7 "Say Verbs" (*declare*); and (4) Class 48.1.2 "Reflexive Verbs of Appearance" (*declare, assert*). The canonical LDOCE codes for each of these four classes, respectively, are: (1) D1 X1 D3 V3 T6 GO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5; (2) D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5; (3) D1-TO T1-TO WV5-TO D5-TO T5-TO T5 I5 X5; and (4) D1-TO T1-TO WV5-TO T1 L1. The largest intersection with the canonical LDOCE codes (see Table 8) occurs with class 37.7 (T5 T1-TO).[17] Thus, Step 6 of the algorithm selects 37.7 as the semantic class for *swear*. (See Table 9.)

Example 2: Semantic classification of *attempt*. The LDOCE specification for this verb is: T1 T3 T4 WV5 N. Using the synonymy feature of WordNet, Step 4 of the classification algorithm automatically extracts five candidate classes associated with the synonyms of this word: (1) Class 29.6 "Masquerade Verbs" (*act*); (2) Class 29.8 "Captain Verbs" (*pioneer*); (3) Class 31.1 "Negative/Positive Amuse Verbs" (*try*); (4) Class 35.6 "Ferret Verbs" (*seek*); and (5) Class 55.2 "Complete Verbs" (*initiate*). The canonical LDOCE codes for each of these five classes, respectively, are: (1) I-AS L9-AS; (2) N D1 X1 I-FOR L9-FOR; (3) D5 I-TO X7 D1-TO T1-TO WV5-TO D1-WITH X7-WITH T1-WITH WV5-WITH X9-WITH D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE I3 T1 L1 WV3 WV4 WV5 WA5; (4) D1-OUT_OF T1-OUT_OF X9-OUT_OF; and (5) T1 L1. The largest intersection with the canonical LDOCE codes (see Table 8) occurs with class 31.1 (WV5). However, Step 5 of the classification algorithm determines that there is a mismatch between the canonical codes for 31.1 and those associated with **both** *try* and *attempt*. In particular, code T3 (see Table 5) emerges as a significant distinction between the two. Thus, a new class (005) is hypothesized with T3 as its canonical LDOCE code (see Tables 10 and 11).[18] Note that this a conservative approach to classification, i.e., rather than assigning a verb to an existing class whose codes are significantly mismatched, the algorithm generates a new class corresponding to the mismatched code. In the case of *attempt*, the conservative approach wins since *attempt* should not be assigned to the "Amuse" class. A side benefit of this step is that the verb *try*, which is **only** classified as an "Amuse" (negative) verb in Levin's system (e.g., *His misbehavior is trying my patience*), is also (correctly) assigned to class 005. Thus, we are able to identify instances of polysemy—both for new verbs (e.g., *bury*, which has two correct assignments in Table 9) and for verbs that already exist in Levin's semantic classes as alternative senses (e.g., *try*).

Example 3: Semantic classification of *calculate*. The LDOCE specification for this verb is: I T1 T5A T6A T6B WV4 WV5. Using the synonymy feature of WordNet, Step 4 of the classification algorithm automatically extracts seven candidate classes associated with the synonyms of this word: (1) Class 13.3 "Verbs of Future Having" (*promise*); (2) Class 26.7 "Performance Verbs" (*direct*); (3) Class 29.1 "Appoint Verbs" (*reckon, designate*); (4) Class 29.2 "Characterize Verbs" (*intend*); (5) Class 29.4 "Declare Verbs" (*judge*); (6) Class 29.5 "Conjecture Verbs" (*figure*); and (7) Class 54.4 "Price Verbs" (*estimate*). The canonical LDOCE codes for each of these seven classes, respectively, are: (1) D1 X1 D1-TO T1-TO WV5-TO; (2) I D1 X1 D1-TO T1-TO WV5-TO T1 L1 T1-FOR D1-FOR; (3) D1 X1 D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T1-AS X9-AS X1-AS; (4) D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T1-AS X9-AS X1-AS; (5) D1 X1 D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5; (6) D3 V3 T6 X1-TO_BE X7-TO_BE X9-TO_BE V3-TO_BE T5 I5 X5; and (7) D1-AT T1-AT X9-AT T1 L1. The largest intersection with the canonical LDOCE codes (see Table 8) occurs with class 29.1 (I T1), but these codes are considered "lower weight" due to their level of generality (see fn. 6). Since class 29.1 contains two matching synonyms and the others only contain one matching synonym, the weighting

heuristic places a higher preference on this class over the others. Thus, Step 6 of the classification algorithm selects 29.1 as the semantic class for *calculate*. As shown in Table 12, this assignment is incorrect; class 54.4 is the class that should have been selected. Hand-verification indicates that the LDOCE code T1-AT is missing from the entry for *calculate*. This code, which **is** included in the entry for the relevant synonym *estimate*, corresponds to a basic construction that applies to verbs in class 54.4, e.g., *They estimated the price of the house at $300K*. Had it been included in the LDOCE entry for *calculate*, this verb would have been correctly classified.

The above experiment indicates that there is a relatively high degree of accuracy in our approach. A larger-scale application of this acquisition technique has allowed us to add 1400 new verbs to Levin's classification. The full set amounts to over 4500 classified verbs, which we we have now distributed among the 192 Levin-based semantic classes plus 26 newly predicted semantic classes. Given that the rules for determining the LCS representation are well-defined (as discussed below), we can automatically generate the LCS representations for each one of these verbs. In the MILT project, we have ported these representations over into the Arabic and Spanish bilingual dictionaries by means of the English gloss.

5.3. LEXICALL: Construction of LCS Representations from Semantic Classes

While Levin's classification provides a unique and extensive catalog of verb classes (such as change of state), it does not define the underlying meaning components of each class. One of the main contributions of the approach described here is that it provides a relation between Levin's classes and meaning components as defined in the LCS representation. For example, we are able to determine whether the verb has an Identificational component of meaning in the LCS simply by checking whether the verb is in the Change of State class. The verbs *break* and *cut* have this component of meaning in the LCS representation constructed by the acquisition procedure, whereas, *hit* and *touch* do not. Figure 10 illustrates the language-independent mapping between verb classes (as delineated by alternations such as Causative and Middle) and the components of meaning associated with the LCS representation for a small set of verbs: *break*, *cut*, *hit*, and *touch*. The top segment of this figure illustrates the assignment of these four verbs into semantic classes based on syntactic alternations, e.g., Conative. The middle segment indicates the rules necessary to map between these classes and the corresponding LCS representations, e.g., 'Change of State → Ident (with state change [such as BROKEN] as result).' The bottom segment shows the LCS representations produced by application of these rules for the four sample verbs.[19]

Our starting point for LEXICALL is the set of English verbs that were assigned to semantic classes by the techniques described in previous sections. This classification covers approximately 4500 English verbs, i.e., a substantial portion of the type of vocabulary of interest in the MILT project. For each class, we have stored a single manually-encoded template based on the mapping in Figure 10. Table 14 shows

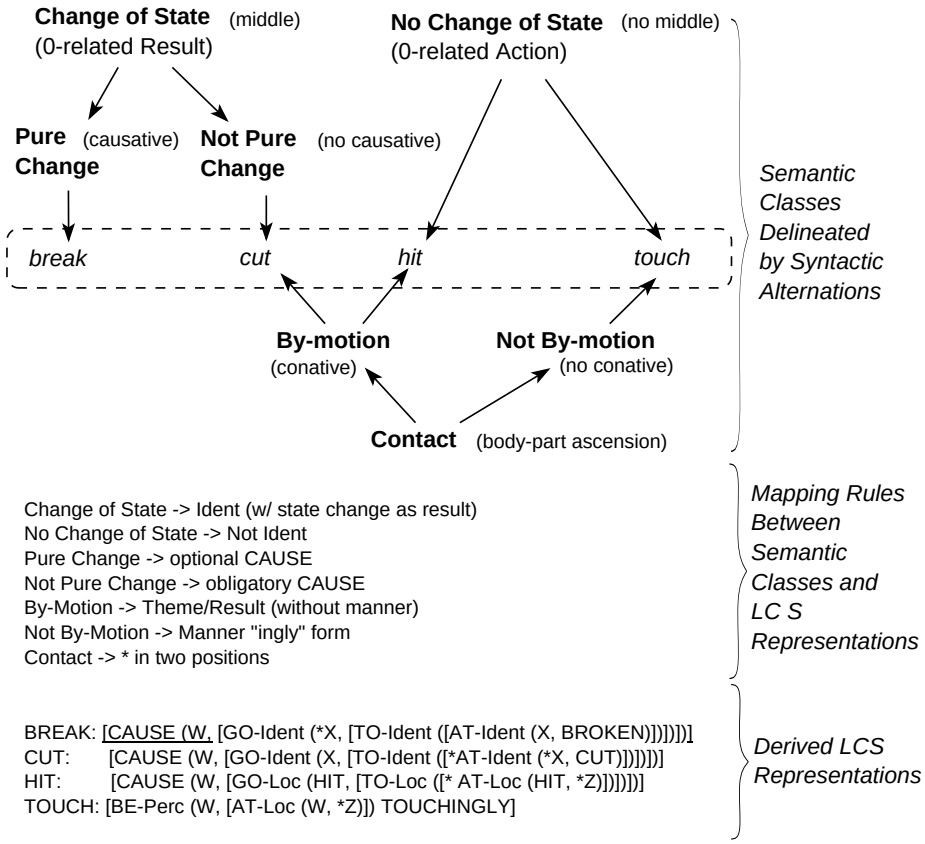

Figure 10. Language-Independent Mapping between Verb Classes and Meaning Components in the LCS

six broad semantic categories and example verbs along with their associated LCS representations. (See (Dorr and Voss, 1996; Dorr and Olsen, 1996) for more details about the Levin-based LCS representations.)

Table 14. LCSs Based on Levin's Verb Classification

Category	**Verb**	**Class**	**LCS**
Motion	leave	51.2	[GO_{Loc} (Y, [⟨DIRECTION⟩$_{Loc}$ (Y, [AT_{Loc} (Y, Z)])])]
	run	51.3.1	[GO_{Loc} (Y, [BY ⟨MANNER⟩])]
Removal	clear	10.3	[CAUSE (X, [GO_{Ident} (Y, [$TOWARD_{Ident}$ (Y, [AT_{Ident} (Y, [⟨STATE⟩$_{Ident}$ ([⟨OF⟩$_{Poss}$ (*HEAD*, Z)])])])])])]
	shovel	10.4.2	[CAUSE (X, [GO_{Ident} (Y)], [BY ⟨MEANS⟩])]
Placement	fill	9.8	[CAUSE (X, [GO_{Ident} (Y, [$TOWARD_{Ident}$ (Y, [AT_{Ident} (Y, [⟨STATE⟩$_{Ident}$ ([⟨WITH⟩$_{Poss}$ (*HEAD*, Z)])])])])])]
	pour	9.5	[CAUSE (X, [GO_{Loc} (Y)], [BY ⟨MANNER⟩])]
Combining	mix	9.8	[CAUSE (X, [GO_{Ident} (Y, [$TOWARD_{Ident}$ (Y, [AT_{Ident} (Y, [⟨STATE⟩$_{Ident}$ ([⟨WITH⟩$_{Poss}$ (*HEAD*, Z)])])])])])]^2
	shake	9.5	[CAUSE (X, [GO_{Loc} (Y)], [BY ⟨MANNER⟩])]
Sound	say	37.7	[CAUSE (X, [GO_{Ident} (Y, [$TOWARD_{Ident}$ (Y, [AT_{Ident} (Y, [⟨STATE⟩$_{Ident}$])])])])]
	shout	37.3	[CAUSE (X, [GO_{Perc} (Y)], [BY ⟨MANNER⟩])]
Killing	kill	42.1	[CAUSE (X, [GO_{Ident} (Y, [$TOWARD_{Ident}$ (Y, [AT_{Ident} (Y, [⟨STATE⟩$_{Ident}$ ([⟨WITH⟩$_{Instr}$ (*HEAD*, Z)])])])])])]
	stab	42.2	[CAUSE (X, [GO_{Ident} (Y)], [BY ⟨MANNER⟩])]

One important benefit of using the Levin classification as the basis of our LCS derivation is that, once all the LCSs for the verbs have been derived, it is not

necessary to store all aspects of verb behavior in the lexical entry. In particular, general principles determine syntactic alternations from this semantic representation. Another benefit of the Levin-based scheme is that, once the mapping between verb classes and LCS representations has been established, we can acquire the LCS representation for a new verb (i.e., one not in Levin) simply by associating it with one of the existing or newly predicted semantic classes.

Another benefit of the LCS is that it is easily ported into new languages using bilingual dictionaries and a condensed representation of the LCS (thematic grids such as ag_th) that are easily verified by a native speaker over a period of approximately two weeks We expect this approach to provide a significantly reduction in lexicon development time as we continue to add more languages to multilingual NLP applications. Dorr, Garman, and Weinberg (1995) estimate that it would take at least 6 months to build such a lexicon from scratch (by human recall and data entry alone), and in such a case, the potential for error would be at least twice as high. This estimate is based on: (1) a manual construction time of 10 minutes per entry (given all the possible usages of a verb) for 4500 verb entries, totaling 5 months, plus (2) an additional month of proofing and consistency checking across entries. Note for example, that because the verbs in LDOCE and WordNet entries are already consistency-checked (through years of revisions by teams of lexicographers), the error rate of the automatically produced lexicons is significantly lower than that of a lexicon constructed from scratch, unless a heavier person-hour commitment can be made to the proofing process.

6. Conclusions and Future Work

We have described techniques for automatic construction of dictionaries for use in large-scale foreign language tutoring and interlingual machine translation systems. The dictionaries are based on a language-independent representation called *lexical conceptual structure*. We started by describing how this representation is used in FLT and MT. Through experimentation, were able to automatically classify a large database of verbs into a rich semantic typology based on (Levin, 1993), from which we can derive the LCS representation.

We have demonstrated that a linguistically motivated technique based on semantic information (from WordNet) and syntactic information (from LDOCE) provides 61% accuracy for classification of 95 unknown verbs. We further suggest that an additional 22% accuracy would be achieved if our online resources (i.e., LDOCE and Levin's patterns) were enhanced to include syntactic information that was discovered to have been omitted. The remaining 17% corresponds to cases where there is a semantic mismatch between WordNet and Levin, i.e., the WordNet synonyms for an unknown verb correspond to word senses that are not available in Levin. These results are summarized as follows:

82	61% Correct Semantic Class Assignments
29	22% Incorrect Based on Syntactic Omissions
23	17% Incorrect Based on Semantic mismatch (WordNet/Levin)
134	100% Total semantic class assignments

Our goal in building the classification algorithm was to show that it is possible to: (1) classify verbs that were not previously defined in Levin's book using the verbs WordNet synonyms and LDOCE codes as a starting point; (2) demonstrate that synonymous verb senses share distributional patterns. With regard to (1), we have shown that correct class assignments can be obtained using these resources two-thirds of the time (for our sample set), without any human intervention. While this result is not as high as we would need for fully automatic semantic classification, it is clearly better to have this as a starting point than to build the semantic classes from scratch—particularly in the cases where new classes are predicted. It is interesting to note that most of the cases where syntactic omissions result in erroneous semantic classifications (see Table 12) are due to missing information in LDOCE. (Such omissions are also described in Boguraev and Briscoe (1989).) Only two cases were attributed to missing patterns in Levin's classes, which perhaps indicates that Levin is more fine-grained with respect to each verb sense (i.e., greater depth than LDOCE), but not as broadly descriptive (i.e., lesser breadth than LDOCE). The prediction of several new semantic classes seems to support this conjecture.

With regard to (2), our results support the existence of a clear relationship between a verb's syntactic behavior and its corresponding word sense(s). In 83% of the cases (61% assigned automatically and 22% assigned following an syntactic enhancements to LDOCE and Levin's patterns), the semantic class assignment was hand-verified to be correct. This means that verbs considered to be synonymous in WordNet do exhibit similar syntactic behaviors. The small residue (17%) includes verbs like *deserve*, which has the WordNet synonym *rate*, but which behaves syntactically more like verbs in the (newly predicted) class of 'Aspire Verbs' (aspire, attempt, deserve, forget, manage, plan, refuse, tend, try, ...).

Two important benefits of this research were: (1) The identification of new semantic classes for verbs that exhibit "non-canonical" syntactic behaviors with respect to existing classes; (2) The assignment of verbs already included in Levin's classes to additional (existing or new) classes. These results contribute both toward the enrichment of existing online resources—LDOCE and (Levin, 1993)—and toward the development of lexicons containing more complete information than is provided in each of these resources alone. The results were made possible by the application of LDOCE-based syntactic constraints *after* a set of class assignments were predicted semantically from WordNet synonyms. The algorithm was designed to detect cases where a new verb (e.g., *attempt*) shares a syntactic code with an already-classified synonym (e.g., *try*) that is not one of the canonical codes for the synonym's class. In such a case, both the verb and its synonym are classified in a newly predicted semantic class (e.g., Class 005 'Aspire Verbs'). The detection of non-canonical syntactic codes contributes to the exhaustiveness of the approach; in contrast to previous approaches (Dorr and Jones, 1996a; Dorr and Jones, 1996b), a semantic

class is assigned even to verbs that do not fit into the existing semantic classes. Moreover, the constraints correspond to both positive and negative sentence patterns, thus imposing stronger restrictions on membership of a verb in a particular semantic class.

While the techniques described above have provided a reasonable foundation for fast semantic classification, we expect that it would be possible to improve the our success rate in future experiments by taking advantage of the separation of word senses in LDOCE for polysemous verbs.[21] In particular, each verb sense in LDOCE is associated with a smaller subset of the full set of syntactic codes that apply to the verb. Currently, our algorithm ignores the division of codes into groups corresponding to separate verb senses, i.e., a single set of LDOCE codes is associated with each English verb, even in cases where a subset of the codes in a pattern corresponds to more than one verb meaning. This is a problem for cases such as the verb *sleep*, which appears in Levin's class 40.4 ("Snooze Verbs") (as in appears in *Gloria slept*) as well as class 54.3 ("Fit Verbs") (as in *The cabin sleeps 5 people*). Both of these senses of *sleep* appear in the LDOCE (the first with the code 'I' and the second with the code 'T1'), yet our current algorithm collapses these two into the same code pattern 'I T1.' Moreover, the algorithm is unable to determine whether there is any connection between such verbs and other verbs that have this code pattern as a sub-pattern (e.g., *nap*, which also occurs in class 40.4 but has the pattern 'I T1 N').

Multiple word meanings potentially lead to a many-to-one mapping—in both directions—between target language items and their English glosses (i.e., many target senses for one English word or one target sense for many English glosses). If one sense maps to many senses in the target language, then the associated LDOCE codes will end up with the wrong semantic mapping and the word will be misclassified. Thus, it is crucial that the codes assigned in the initial language (English, in this case) be correctly separated into groups that correspond to independent word meanings. In future experiments, we will be separating LDOCE according to word meaning prior to activation of the acquisition program; we expect that this will significantly reduce the need for post hoc human verification.

The underspecificity of certain LDOCE codes also requires further investigation. The syntactic tests that Levin uses to distinguish semantic classes are frequently much more subtle than the syntactic subcategorization information available in the LDOCE. For example, two crucial syntactic tests that Levin uses for distinguishing semantic classes are the Unspecified Object Alternation (e.g., that it is possible to say *John ate* as well as *John ate the apple*) and the Causative/Inchoative alternation (e.g., that it is possible to say *She broke the vase* as well as *The vase broke*). However, these two tests are collapsed into a single LDOCE code pair T1/I in our system. Thus, when our algorithm accesses the LDOCE codes, the verbs *eat* and *break* are assumed to have identical syntactic behaviors.[22] This type of collapsing is generally incorrect because, for example, the code 'I' might stand for an agent (as in *John ate* in one case, but a theme in another (as in *The window broke*).[23]

A future enhancement to the system would be to analyze dictionary definitions associated with verbs in the LDOCE. In particular, if we search the text of dictionary definitions for key words and phrases, it might be possible to pick out information that would help distinguish between two alternations. For example, verbs in the "Change of State" class which have the T1/I encoding (e.g., *break*), often have the word *cause* as part of their dictionary definition; the presence of *cause* indicates that T1/I should be correlated with the Causative/Inchoative alternation, not the Unspecified Object alternation. An analysis of the entry The verb *break* is defined: "to cause to separate into parts suddenly or violently." The verb *eat*, which also has the T1/I encoding, on the other hand, is defined as follows: "to take in through the mouth and swallow." Thus, it is possible to distinguish between the two possible correlations associated with the T1/I encoding. An attempt to pick out verbs in the "Change of State" class using a similar technique has been proposed by Fontenelle and Vanandroye (1989). A preliminary investigation indicates that the LDOCE would be useful in this regard as well.

It should be noted that the importation of LCSs from English glosses into other languages is viewed as a first approximation to complete lexicons for these languages. The results must still be hand-checked by native speakers using a readable representation for each semantic class. Even so, building such lexicons from scratch would take 12 times the amount of time that it takes to verify the acquisition results by hand (approximately two weeks). Thus, we expect our automatic acquisition techniques to improve LCS-based lexicon development time significantly as we continue to add more languages to multilingual NLP applications.

Acknowledgments

A special thanks goes to Mari Olsen for comments and thorough proofreading of this document. I would also to thank Scott Blanksteen, Jim Hendler, Doug Jones, Boyan Onyshkevich, Jungshin Park, Ulku Sencan, Clare Voss, and Amy Weinberg, for helpful suggestions during the investigation reported herein. Finally, I am indebted to anonymous reviewers who helped me shape this article into its final form. This work was supported, in part, by Department of Defense contract MDA90496C1250, Army Research Office contract DAAL03-91-C-0034 through Battelle Corporation, NSF NYI IRI-9357731 and Logos Corporation, NSF CNRS INT-9314583, Advanced Research Projects Agency and ONR contract N00014-92-J-1929, Alfred P. Sloan Research Fellow Award BR3336, Army Research Institute contract MDA-903-92-R-0035 and Microelectronics and Design, Inc., and the University of Maryland General Research Board.

Notes

1. The first languages targeted for this system are Arabic and Spanish.

2. One reviewer commented that the approach described here appears to take an 'opposite' view from that of Levin (1993), where a verb's behavior is said to be fully semantically determined. We do not see the two views to be in opposition. Levin provides a catalog of verb behaviors and assigns verbs that behave similarly to named classes. Frequently, the classes (or the choices of relevant syntactic behavior) have names that were inspired by semantics, but the basic approach is to examine syntactic behavior. The approach described here extends Levin's work, relying on her central thesis concerning the syntax/semantics relation—verified by Dorr and Jones (1996b), which paid close attention to the problem of polysemy—and using the database for the purposes of constructing a lexicon. As will be demonstrated shortly, Levin's classes are not considered to be final; we have added several new classes to create LCSs for even the verbs just in (Levin, 1993). In addition, more recent work by Dorr and Olsen (1997) has suggested that approximately 400 classes (and perhaps more) are needed to capture aspectual facts.
3. Yarowsky's work focuses on nouns and, as such, is complementary to our work, which emphasizes verb classification. A fruitful area of future investigation would be a combination of the two approaches, perhaps applying statistical techniques for disambiguation of nouns, coupled with access to higher-level syntactic information for disambiguation of verbs. Combining two different approaches is supported by the work of Fisher et al. (1994) which demonstrates that nouns are acquired by pairing each noun with a world concept in the world, whereas verbs are acquired through access to the syntactic structures associated with each verb.
4. Detection of the measure phrase by a simple syntactic parser is a non-trivial problem. See related discussion below in (11).
5. The final four entries in Table 3 correspond to alternations that were not previously listed in (10) above; these are:

- **Locative:**
  John loaded hay onto the truck
  John loaded the truck with hay
- **Middle:**
  Mary cut the fabric
  The fabric cuts easily
- **Causative:**
  Mary lay the books on the table
  The books lay on the table
- **Dative:**
  John sent a package to Mary
  John sent Mary a package

   Note, for example, that verbs in class 9.2 *Put in a Spatial Configuration* (e.g., *lie*) participate in both alternates of the Causative pair.

6. The Spanish-English dictionary was built at the University of Maryland as a part of the MILT project. The Arabic-English dictionary was produced by Alpnet, a company in Utah that develops translation aids. We are also in the process of developing bilingual dictionaries for Korean and French, and we will be porting our LCS acquisition technology to these languages in the near future.
7. Non-English verbs were initially assigned to semantic classes by using the English classification in conjunction with a bilingual lexicon. A native speaker verified the resulting assignments by a process that took approximately 2 weeks per language (Dorr, Garman, and Weinberg, 1995). This verified database was provided as input to the LEXICALL program, which produced the corresponding LCSs by means of the same steps that were applied as it did for the English verbs. To compensate for cross-language distinctions, LEXICALL was designed to take input that includes other phrasal units (from the bilingual lexicon), where necessary, e.g., the Spanish equivalent of the English verb *enter* includes a preposition: *entrar a*. In such cases, LEXICALL incorporated this additional information into the LCS entry.

8. We use the term MRD to refer any online, lexical database, including monolingual dictionaries, bilingual dictionaries, and thesaurus-like resources.
9. It is possible that this was not yet known at the time of the study by Boguraev and Briscoe (1989), given that (Levin, 1993) was published 3 years later.
10. A hint is given (p. 143) that future investigation is headed in the direction of automatic extraction of "case roles", but no details are given.
11. Both the database of sentences and the syntactic pattern extraction program are encoded in Quintus Prolog. These patterns were used in previous work (Dorr and Jones, 1996a; Dorr and Jones, 1996b) which assigned verbs to semantic classes by first examining syntactic information (LDOCE codes) and then applying a semantic filter (based on WordNet). Application of the semantic filter *after* syntactic codes were accessed frequently resulted in the elimination of all possible assignments for a given verb. The approach described herein operates from the opposite perspective: rather than applying a WordNet-based filter to syntactically induced class assignments, it examines WordNet synonyms and then applies a syntactic LDOCE-based filter to rule out illegal class assignments. The syntactic filter is tighter than that of the previous approach, using both positive and negative constraints—legal and illegal sentence patterns, respectively. At the same time, it is more exhaustive than the previous approach in that it produces a semantic class assignment for *all* verbs, even those that do not fit into existing semantic classes. This is achieved by assigning new semantic classes to verbs that do not adhere to syntactic constraints imposed by existing classes.
12. When a new class is hypothesized for V, the synonyms of V which already occur in a different Levin class are also placed in this new class. For example, the verb *try*—which only occurs in the semantic class of "Negative Amuse" verbs (as in *try one's patience*) in (Levin, 1993)—is classified into an additional (new) class when it is discovered to be synonymous with a verb not in (Levin, 1993), *attempt*. We will see how this case is handled shortly.
13. The weighting heuristic refers to a preference for those classes containing the highest number of matching WordNet synonyms. For example, the semantic classes whose canonical LDOCE codes match the codes for the verb *reduce* are: 22.4 (*trammel*), 10.4.1 (*trim*), 31.1 (*cut*), 21.1 (*cut*), and 26.6 (*turn, change*). Because Class 26.6 "Turn Verbs" contains two matching WordNet synonyms and the others contain one, this class is selected for *reduce*. (This choice has been verified to be correct.) When two or more classes contain the highest number of matching synonyms, all such classes are selected.
14. We originally had 114 control vocabulary words. However, several of these had WordNet synonym sets that did not intersect with verbs in the Levin-based semantic classes. In later iterations of the acquisition program we were able to classify the remaining 9 control verbs by using classes that were acquired in previous iterations. For example, the verb *trick* was not classifiable initially because its synonyms (e.g., *deceive*) were not yet in Levin's original classes. However, once the verb *deceive* was classified into semantic class 33, we re-ran the program to classify the verb *trick* appropriately.
15. The only exception to this is the verb *help*, which fortuitously was assigned to the correct class (13.4.1), and no others, by the weighting heuristic. The bracketed code T1-WITH is listed with the verb *help* only because it is a canonical LDOCE code for class 13.4.1 and it was found to be relevant to the verb, e.g., *She helped him with his homework.*
16. In such cases, we restricted our hand corrections to those new classes identified by the algorithm; we did not hand generate any new classes.

17. This example illustrates an important point that was not described above. If the LDOCE codes are taken in raw form, the intersection of codes associated with class 48.1.2 (T1 T1-TO) would be the same length as that of class 37.7 (T5 T1-TO). But the codes are given different weights; in particular, T1 is one of three codes that is given a lower weight since it refers to a very common syntactic frame, i.e., the bare transitive. Two additional lower-weighted codes are I (bare intransitive, e.g., *sleep*) and N (verb has zero-related nominal, e.g., *shovel*). These codes were found to provide very little information toward distinguishing among semantic classes, and thus were ignored unless no other distinguishing information was available. Note: This type of "weighting" is different from the heuristic referred to earlier, which induces a preference for those classes whose canonical LDOCE codes match a high number of WordNet synonyms. A later example illustrates the application of this heuristic.
18. This new class was later hand-labeled "Aspire Verbs."
19. The underlined LCS components (e.g., in the entry for *break*) are intended to be optional, resulting from the application of the mapping rule 'Pure Change → optional CAUSE.'
20. The *HEAD* symbol is a place-holder that points to the root (CAUSE) of the overall lexical entry. Modifiers, such as instrumental phrases, typically include this symbol. See (Dorr et al., 1993) for more details of the LCS notation.
21. The idea of sense disambiguation was also central in the work of (Dorr and Jones, 1996b) where a significant improvement in acquisition was achieved by dividing the syntactic cues into distinct groupings that correlated with different word senses in (Levin, 1993).
22. It is for this reason that T1 and I are among the "lower weight" codes, as described earlier in Example 3.
23. Fillmore (1970) argues that these two types of intransitives must crucially be viewed as distinct constructions in order to distinguish semantically between verbs like *hit* and *break*.

References

Alshawi, H. 1989. Analysing the Dictionary Definitions. In B. Boguraev and T. Briscoe, editors, *Computational Lexicography for Natural Language Processing.* Longman, London, pages 153–169.

Ballard, B. and D.E. Stumberger. 1986. Semantic Acquisition in TELI. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 20–29.

Barnett, Jim, Kevin Knight, Inderjeet Mani, and Elaine Rich. 1990. Knowledge and Natural Language Processing. *Communications of the ACM*, pages 50–71, August.

Bates, Madelyn and R. Bobrow. 1983. Information Retrieval Using a Transportable Natural Language Interface. In *Proceedings of the Sixth Annual ACM SIG Conference on Research and Development in Information Retrieval*, pages 81–86.

Boguraev, Branimir and Ted Briscoe. 1989. Utilising the LDOCE Grammar Codes. In Branimir Boguraev and Ted Briscoe, editors, *Computational Lexicography for Natural Language Processing.* Longman, London, pages 85–116.

Brent, Michael. 1993. Unsupervised Learning of Lexical Syntax. *Computational Linguistics*, 19:243–262.

Carrier, Jill and Janet H. Randall. 1993. Lexical mapping. In Eric Reuland and Werner Abraham, editors, *Knowledge and Language II: Lexical and Conceptual Structure*. Kluwer, Dordrecht, pages 119–142.

Chomsky, Noam. 1981. *Lectures on Government and Binding*. Foris Publications, Dordrecht, Holland.

Chomsky, Noam. 1986. *Knowledge of Language: Its Nature, Origin and Use*. The MIT Press, Cambridge, MA.

Church, Kenneth and P. Hanks. 1990. Word Association Norms, Mutual Information and Lexicography. *Computational Linguistics*, 16:22–29.

Copestake, Ann, Ted Briscoe, P. Vossen, A. Ageno, I. Castellon, F. Ribas, G. Rigau, H. Rodríguez, and A. Samiotou. 1995. Acquisition of Lexical Translation Relations from MRDS. *Machine Translation*, 9:183–219.

Corbin, Wilbur, D. Copeland, and Brian Buck. 1994. Determining Verb Usage from Parsed Corpora: Matrix of Levin's Syntactic/Semantic Classes. Technical Report Project Report for NLP Course (CMSC 723), University of Maryland, College Park, MD.

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

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

Dorr, Bonnie J. 1994. Machine Translation Divergences: A Formal Description and Proposed Solution. *Computational Linguistics*, 20(4):597–633.

Dorr, Bonnie J. To appear in 1997. Large-Scale Acquisition of LCS-Based Lexicons for Foreign Language Tutoring. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP)*, Washington, DC.

Dorr, Bonnie J., Joseph Garman, and Amy Weinberg. 1995. From Syntactic Encodings to Thematic Roles: Building Lexical Entries for Interlingual MT. *Machine Translation*, 9:71–100.

Dorr, Bonnie J., James Hendler, Scott Blanksteen, and Barrie Migdalof. 1993. Use of Lexical Conceptual Structure for Intelligent Tutoring. Technical Report UMIACS TR 93-108, CS TR 3161, University of Maryland.

Dorr, Bonnie J., Jim Hendler, Scott Blanksteen, and Barrie Migdalof. 1995. Use of LCS and Discourse for Intelligent Tutoring: On Beyond Syntax. In Melissa Holland, Jonathan Kaplan, and Michelle Sams, editors, *Intelligent Language Tutors: Balancing Theory and Technology*. Lawrence Erlbaum Associates, Hillsdale, NJ, pages 289–309.

Dorr, Bonnie J. and Douglas Jones. 1996a. Acquisition of Semantic Lexicons: Using Word Sense Disambiguation to Improve Precision. In *Proceedings of the Workshop on Breadth and Depth of Semantic Lexicons, 34th Annual Conference of the Association for Computational Linguistics*, pages 42–50, Santa Cruz, CA.

Dorr, Bonnie J. and Douglas Jones. 1996b. Role of Word Sense Disambiguation in Lexical Acquisition: Predicting Semantics from Syntactic Cues. In *Proceedings of the International Conference on Computational Linguistics*, pages 322–333, Copenhagen, Denmark.

Dorr, Bonnie J., Jye-hoon Lee, Clare Voss, and Sungki Suh. 1995a. Development of Interlingual Lexical Conceptual Structures with Syntactic Markers for Machine Translation. Technical Report UMIACS TR 95-16, CS TR 3412, University of Maryland, College Park, MD.

Dorr, Bonnie J., Dekang Lin, Jye-hoon Lee, and Sungki Suh. 1995b. Efficient Parsing for Korean and English: A Parameterized Message Passing Approach. *Computational Linguistics*, 21(2):255–263.

Dorr, Bonnie J. and Mari Broman Olsen. 1996. Multilingual Generation: The Role of Telicity in Lexical Choice and Syntactic Realization. *Machine Translation*, 11(1–3):37–74.

Dorr, Bonnie J. and Mari Broman Olsen. 1997. Aspectual Modifications to a LCS Database for NLP Applications. Technical Report LAMP TR 007, UMIACS TR 97-23, CS TR, University of Maryland, College Park, MD.

Dorr, Bonnie J. and Clare Voss. 1996. A Multi-Level Approach to Interlingual MT: Defining the Interface between Representational Languages. *International Journal of Expert Systems*, 9(1):15–51.

Farwell, David, Louise Guthrie, and Yorick Wilks. 1993. Automatically Creating Lexical Entries for ULTRA, a Multilingual MT System. *Machine Translation*, 8(3):127–145.

Fillmore, Charles. 1968. The Case for Case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart, and Winston, pages 1–88.

Fillmore, Charles J. 1970. The Grammar of Hitting and Breaking. In R.A. Jacobs and P.S. Rosenbaum, editors, *Readings in English Transformational Grammar*. Ginn Pubs., Waltham, MA, pages 120–133.

Fisher, Cynthia, Henry Gleitman, and Lila Gleitman. 1991. On the Semantic Content of Subcategorization Frames. *Cognitive Psychology*, 23(3):331–392.

Fisher, Cynthia, D. Geoffrey Hall, Susan Rakowitz, and Lila Gleitman. 1994. When it is better to receive than to give: Syntactic and conceptual constraints on vocabulary growth. In L. Gleitman and B. Landau, editors, *The Acquisition of the Lexicon*. Lingua 92, The MIT Press, Cambridge, MA, pages 333–375.

Foley, William A. and Robert D. Van Valin. 1984. *Functional Syntax and Universal Grammar*. Cambridge University Press, Cambridge.

Fontenelle, T. and J. Vanandroye. 1989. Retrieving Ergative Verbs from a Lexical Data Base. *Dictionaries*, 11:11–39.

Ginsparg, J. 1983. A Robust Portable Natural Language Data Base Interface. In *Proceedings of the Conference on Applied Natural Language Processing*.

Gleitman, Lila. 1990. The Structural Sources of Verb Meanings. *Language Acquisition*, 1(1):3–55.

Grimshaw, Jane. 1990. *Argument Structure*. The MIT Press, Cambridge, MA.

Grimshaw, Jane. 1994. Semantic Structure and Semantic Content in Lexical Representation. unpublished ms., Rutgers University, New Brunswick, NJ.

Grishman, Ralph. 1986. *Computational Linguistics: An Introduction*. ACL Studies in Natural Language Processing. Cambridge University Press.

Grosz, Barbara J., Douglas E. Appelt, Paul. A. Martin, and Fernando C.N. Pereira. 1987. TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces. *Artificial Intelligence*, 32(2):173–243.

Gruber, Jeffrey S. 1965. *Studies in Lexical Relations*. Ph.D. thesis, MIT, Cambridge, MA.

Guida, G. and C. Tasso. 1983. IR-NL1: An Expert Natural Language Interface to Online Data Bases. In *Proceedings of the Conference on Applied Natural Language Processing*.

Hale, Ken and Samuel J. Keyser. 1993a. On Argument Structure and Lexical Expression of Syntactic Relations. In Ken Hale and Samuel J. Keyser, editors, *The View from Building 20: Essays in Honor of Sylvain Bromberger*. The MIT Press, Cambridge, MA, pages 53–59.

Hale, Ken and Samuel J. Keyser. 1993b. *On Argument Structure and the Lexical Expression of Syntactic Relations*. The MIT Press, Cambridge, MA.

Hogan, Chris and Lori Levin. 1994. Data Sparseness in the Acquisition of Syntax-Semantics Mappings. In *Proceedings of the Post-COLING94 International Workshop on Directions of Lexical Research*, pages 153–159, Nicoletta Calzolari and Chengming Guo (co-chairs), Tshinghua University, Beijing.

Holland, Melissa. 1994. Intelligent Tutors for Foreign Languages: How Parsers and Lexical Semantics can Help Learners and Assess Learning. In *Proceedings of the Educational Testing Service Conference on Natural Language Processing Techniques and Technology in Assessment and Education*, Princeton, NJ: ETS.

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

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

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

Jones, Douglas, Robert Berwick, Franklin Cho, Zeeshan Khan, Karen Kohl, Naoyuki Nomura, Anand Radhakrishnan, Ulrich Sauerland, and Brian Ulicny. 1994. Verb Classes and Alternations in Bangla, German, English, and Korean. Technical report, Massachusetts Institute of Technology.

Kemmer, Suzanne, editor. 1993. *The Middle Voice.* John Benjamins, Amsterdam.

Klavans, Judith L. and Evelynne Tzoukermann. 1995. Dictionaries and Corpora: Combining Corpus and Machine-Readable Dictionary Data for Building Bilingual Lexicons. *Machine Translation*, 10:185–218.

Knight, Kevin. 1991. *Integrating Knowledge Acquisition and Language Acquisition.* Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. CMU-CS-91-209.

Landau, Barbara and Lila Gleitman, editors. 1985. *Language and Experience: Evidence from the Blind Child.* Harvard University Press, Cambridge, MA.

Lenat, Douglas B. and R.V. Guha. 1990. *Building Large Knowledge-Based Systems.* Reading, MA, Addison-Wesley.

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

Levin, Beth and Malka Rappaport Hovav. To appear. Building Verb Meanings. In M. Butt and W. Gauder, editors, *The Projection of Arguments: Lexical and Syntactic Constraints.* CSLI.

Lonsdale, Deryle, Teruko Mitamura, and Eric Nyberg. 1995. Acquisition of Large Lexicons for Practical Knowledge-Based MT. *Machine Translation*, 9:251–283.

Miller, George A. 1986. Dictionaries in the Mind. *Language and Cognitive Processes*, 1:171–185.

Miller, George A. 1990. WordNet: An On-Line Lexical Database. *International Journal of Lexicography*, 3:235–312.

Miller, George A. and Christiane Fellbaum. 1991. Semantic Networks of English. In Beth Levin and Steven Pinker, editors, *Lexical and Conceptual Semantics, Cognition Special Issue.* Elsevier Science Publishers, B.V., Amsterdam, The Netherlands, pages 197–229.

Mitamura, Teruko. 1990. *The Hierarchical Organization of Predicate Frames for Interpretive Mapping in Natural Language Processing.* Ph.D. thesis, Department of Linguistics, University of Pittsburgh, Pittsburgh, PA.

Neff, Mary and Michael McCord. 1990. Acquiring Lexical Data from Machine-Readable Dictionary Resources for Machine Translation. In *Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages (TMI-90)*, pages 85–90, Austin, Texas.

Olsen, Mari Broman. To appear in 1997. *The Semantics and Pragmatics of Lexical and Grammatical Aspect.* Garland, New York.

Pesetsky, David. 1982. *Paths and Categories.* Ph.D. thesis, MIT, Cambridge, MA.

Pinker, Steven. 1989. *Learnability and Cognition: The Acquisition of Argument Structure.* The MIT Press, Cambridge, MA.

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

Pugeault, F., P. Saint-Dizier, and M.G. Monteil. 1994. Knowledge Extraction From Texts: A Method For Extracting Predicate-Argument Structures From Texts. In *Proceedings of Fifteenth International Conference on Computational Linguistics*, pages 1039–1043, Kyoto, Japan.

Saint-Dizier, Patrick. 1996a. Constructing Verb Semantic Classes in French: Methods and Associated Semantic Representations. In *Proceedings of the International Conference on Computational Linguistics*, pages 1127–1130, Copenhagen, Denmark.

Saint-Dizier, Patrick. 1996b. Semantic Verb Classes Based on 'Alternations' and on WordNet-like Semantic Criteria: A Powerful Convergence. In *Proceedings of the Workshop on Predicative Forms in Natural Language and Lexical Knowledge Bases*, pages 62–70, Toulouse, France.

Sams, Michelle. 1995. Advanced Technologies for Language Learning: The BRIDGE Project Within the ARI Language Tutor Program. In Melissa Holland, Jonathan Kaplan, and Michelle Sams, editors, *Intelligent Language Tutors: Theory Shaping Technology.* Lawrence Erlbaum Associates, Hillsdale, NJ, pages 7–21.

Sanfilippo, Antonio and V. Poznanski. 1992. The Acquisition of Lexical Knowledge from Combined Machine-Readable Dictionary Resources. In *Proceedings of the Applied Natural Language Processing Conference*, pages 80–87, Trento, Italy.

Templeton, M. and J. Burger. 1983. Problems in Natural Language Interface to DBMS with Examples for EUFID. In *Proceedings of the Conference on Applied Natural Language Processing*, pages 3–16.

Thompson, B. and F. Thompson. 1983. Introducing ASK, a Simple Knowledgeable System. In *Proceedings of the Conference on Applied Natural Language Processing*, pages 17–24.

Walker, Donald and Robert Amsler. 1986. The Use of Machine-readable Dictionaries in Sublanguage Analysis. In R. Grishman and R. Kittredge, editors, *Analyzing Language in Restricted Domains*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, pages 69–83.

Weinberg, Amy, Joseph Garman, Jeffery Martin, and Paola Merlo. 1995. Principle-Based Parser for Foreign Language Training in German and Arabic. In Melissa Holland, Jonathan Kaplan, and Michelle Sams, editors, *Intelligent Language Tutors: Theory Shaping Technology*. Lawrence Erlbaum Associates, Hillsdale, NJ, pages 23–44.

Wilks, Yorick, Dan Fass, Cheng-Ming Guo, James E. McDonald, Tony Plate, and Brian M. Slator. 1989. A Tractable Machine Dictionary as a Resource for Computational Semantics. In Branimir Boguraev and Ted Briscoe, editors, *Computational Lexicography for Natural Language Processing*. Longman, London, pages 193–228.

Wu, D. and X. Xia. 1995. Large-Scale Automatic Extraction of an English-Chinese Translation Lexicon. *Machine Translation*, 9:285–313.

Yarowsky, David. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annaul Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, MA.