LEXICALL: Lexicon Construction for Foreign Language Tutoring

Bonnie J. Dorr
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, USA
bonnie@cs.umd.edu

Abstract: We focus on the problem of building large repositories of *lexical conceptual structure* (LCS) representations for verbs in multiple languages. One of the main results of this work is the definition of a relation between broad semantic classes and LCS meaning components. Our acquisition program—LEXICALL—takes, as input, the result of previous work on verb classification and thematic grid tagging, and outputs LCS representations for different languages. These representations have been ported into English, Arabic and Spanish lexicons, each containing approximately 9000 verbs. We are currently using these lexicons in an operational foreign language tutoring and machine translation.

1 Introduction

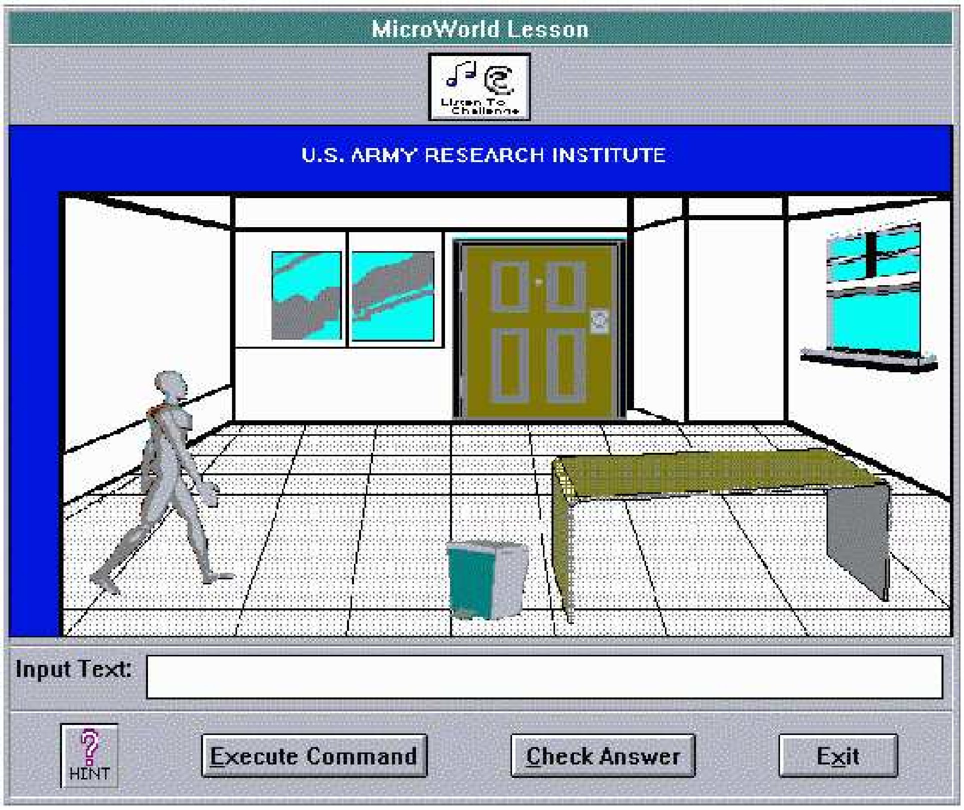
A wide range of new capabilities in NLP applications has been made possible by recent advances in lexical semantics (Carrier and Randall, 1993; Fillmore, 1968; Foley and Van Valin, 1984; Grimshaw, 1990; Gruber, 1965; Hale and Keyser, 1993; Jackendoff, 1983; Jackendoff, 1990; Jackendoff, 1996; Levin, 1993; Levin and Rappaport Hovav, To appear; Pesetsky, 1982; Pinker, 1989). Many of these researchers adopt the hypothesis that verbs can be grouped into broad classes, each of which corresponds to some combination of basic meaning components. This is the basic premise underlying our approach to multilingual lexicon construction. In particular, we have organized verbs into broad semantic classes and subsequently designed a language-independent representation, *lexical conceptual structure* (LCS), for each class; these representations have been ported into Arabic and Spanish on a per-class basis. The result is a large multi-lingual repository of verb entries: our Arabic and Spanish lexicons contain approximately 40,000 Arabic and Spanish entries each, of which 9,000 (per language) are verbs.

We are currently using these lexicons in an operational foreign language tutoring (FLT) system called Military Language Tutor (MILT). This system provides a wide range of lessons for use in language training. Because of the multilingual nature of the system (i.e., broad coverage of Arabic and Spanish), the representations we use must necessarily have crosslinguistic validity as well as coverage for diverse constructions that are likely to occur in lessons.

One of the tutoring lessons, the MicroWorld Lesson (see Figure 1) requires the capability for the language-learner to state domain-specific actions in a variety of different ways. For example, the language-learner might command the agent (pictured at the left in the graphical interface) to take the following action: *Walk to the table and pick up the document.* The same action should be taken if the user says: *Go to the table and remove document*, *Retrieve the document from the table*, etc. The LCS representation provides the capability to execute various forms of the same command without hardcoding them as part of the graphical interface.

In another tutoring lesson, Question-Answering, the student is asked to answer questions about a foreign language text that they have read. Their answer is converted into an LCS which is matched against a prestored LCS corresponding to an answer typed in by a human instructor (henceforth, called the "author"). The prestored LCS is an idealized form of the answer to a question, which can take one of many forms. Suppose, for example, the question posed to the user is: *Where did Jack put the book?* (or *Adónde puso Jack el libro?* in Spanish). The author's answer, e.g., *Jack put the book in the trash*, has been stored as an LCS by the tutoring system. If the student types *Jack threw the book in the trash* or *Jack moved the book from the table into the trash*, the system is able to match against the prestored LCS and determine that all three of these responses are semantically appropriate.

Figure 1: MicroWorld Lesson in MILT



We have developed an acquisition program—LEXICALL—that allows us to construct LCS-based lexicons for the FLT system. LEXICALL is designed so that it can be used for multiple languages, and also for additional NLP applications (e.g., machine translation). One of the main results of this work is the definition of a relation between broad semantic classes (based on work by Levin (1993)) and LCS meaning components. The program builds on previous work, where verbs were classified automatically (Dorr and Jones, 1996; Dorr, To appear) and tagged with thematic grid information (Dorr et al., 1995). We use these pre-assigned classes and thematic grids as input to LEXICALL. The output is a set of LCS's corresponding to individual verb entries in our lexicon.

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). Others have taken a more knowledge-based (interlingual) approach (Lonsdale et al., 1995). Still others, e.g., Copestake et al. (1995), use English-based grammatical codes for acquisition of lexical representations.

Our approach differs from these in that it exploits certain linguistic constraints that govern the relation between a word's surface behavior and its corresponding semantic class. We demonstrate that a LCS representation can be assigned to each semantic class. These, in turn, are ported into multiple languages. We first show how the LCS is used in a FLT system. We then describe the techniques used for automatic acquisition of LCS-based lexicons.

2 Application of the LCS Representation to FLT

One of the types of knowledge that must be captured in FLT 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 *Mary* is the agent in *Mary hit the ball*), and lexical-semantic information (e.g., 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.

This section describes the use of the LCS representation in a question-answering component of the MILT system (Sams, 1993; Weinberg et al., 1995). As described above, 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 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 a match against the corresponding prestored LCS representation. The student is then informed as to whether the question has been answered correctly depending on how closely the student's response LCS matches the author's prestored LCS.

Table 1: Correspondence Between NLP Output and Tutor Feedback

System Prompt: Where did Jack put the book?			
Student Answer	**Prestored Answer**	**Matcher Output**	**Feedback**
Jack threw the book in the trash	Jack threw the book in the trash	exact match	"That's right"
Jack put the book in the trash	Jack threw the book in the trash	missing MANNER	"How?"
Jack threw the book in the trash	Jack put the book in the trash	extra MANNER	"You're assuming things"
Jack is friendly	Jack put the book in the trash	mismatch primitive	"Please reread"
Jack threw the book	Jack put the book in the trash	missing argument	"Where?"

Consider what happens in a lesson if the author has specified that a correct answer to the question *Adónde puso Jack el libro?* in Spanish is *Jack tiró el libro a la basura* ('Jack threw out the book into the trash'). This answer is processed by the system to produce the following LCS:

(1) $[_{\text{Event}}\ \text{CAUSE}$
$\quad ([_{\text{Thing}}\ \text{JACK}],$
$\quad\ [_{\text{Event}}\ \text{GO}_{\text{Loc}}$
$\quad\quad ([_{\text{Thing}}\ \text{BOOK}],$
$\quad\quad\ [_{\text{Path}}\ \text{TO}_{\text{Loc}}$
$\quad\quad\quad ([_{\text{Position}}\ \text{AT}_{\text{Loc}}\ ([_{\text{Thing}}\ \text{BOOK}],\ [_{\text{Thing}}\ \text{TRASH}])])])],$
$\quad\ [_{\text{Manner}}\ \text{THROWINGLY}])]$

The LCS is stored by the tutor and then later matched against the student's answer. If the student types *Jack movio' el libro de la mesa a la basura* ('Jack moved the book from the table to the trash'), the system must determine if these two match. The student's sentence is processed and the following LCS structure is produced:

(2) $[_{\text{Event}}\ \text{CAUSE}$
$\quad ([_{\text{Thing}}\ \text{JACK}],$

$[_{\text{Event}}\ \text{GO}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{BOOK}],$
$[_{\text{Path}}\ \text{TO}_{\text{Loc}}\ ([_{\text{Position}}\ \text{AT}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{BOOK}],\ [_{\text{Thing}}\ \text{TRASH}])])],$
$[_{\text{Path}}\ \text{FROM}_{\text{Loc}}\ ([_{\text{Position}}\ \text{AT}_{\text{Loc}}$
$([_{\text{Thing}}\ \text{BOOK}],\ [_{\text{Thing}}\ \text{TABLE}])])])])]$

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

```
Missing: MANNER THROWINGLY
Extra: FROM LOC
```

The system identifies the student's response as a match with the prestored answer, but it also recognizes that there is one piece of missing information and one piece of extra information.

The "Missing" and "Extra" output is internal to the NLP component of the Tutor, i.e., this 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 that was originally given. For example, the instructor can program the tutor to notify the student about the omitted information in the form of a 'How' question, or it can choose to ignore it. The extra information is generally ignored, although it is recorded in case the instructor decides to program the system to notify the student about this as well. The full range of feedback is not presented here. Some possibilities are summarized (in English) in Table 1 (adapted from (Holland, 1994)). Note that the main advantage of using the LCS is that it allows the author to type in an answer that is general enough to match any number of additional answers.

3 Overview of LCS Acquisition

Our work on construction of LCS lexicons has focused primarily on building entries for verbs.[1] This section outlines our approach to verb acquisition.

LEXICALL requires three inputs for acquisition of verb entries: a semantic class, a thematic grid, and a lexeme, which we will henceforth abbreviate as "class/grid/lexeme". An example of input/output for the system is shown here:

(3) **Acquisition of LCS for:** *touch*
Input: 47.8; `_th_loc`; "touch"
Output:
`(be loc (* thing 2) (at loc (thing 2) (* thing 11)) (touchingly 26))`

[1] However, we have approximately 30,000 non-verb entries per language.

The lexeme is, by convention, an English-like word whose morphological variant ultimately fills some position in the LCS definition. The verb that will be associated with the resulting LCS is not necessarily the same as this lexeme. For example, the Spanish counterpart of *touch*, also uses the "touch" lexeme.[2]

The semantic class label 47.8 above is taken from Levin's 1993 book (Verbs of Contiguous Location), i.e., the class to which the verb *touch* has been assigned.[3] The thematic grid (`_th_loc`) is taken from a set of grid assignments previously assigned by (Dorr et al., 1995). The following conventions are adopted for thematic grid specifications: (1) Any thematic role preceded by an underscore (_) is obligatory; (2) Any thematic role preceded by a comma (,) is optional; (3) Any thematic role that is followed by a parenthesized preposition must necessarily be contained in a phrase headed by that preposition.

Table 2: Sample Templates Stored in the LCS Database

Category	**Verb**	**Class**	**LCS**
Location	suspend	9.2	$[\mathrm{CAUSE}\ (\mathrm{X},\ [\mathrm{BE}_{\mathrm{Loc}}\ (\mathrm{Y},\ [\mathrm{AT}_{\mathrm{Loc}}\ (\mathrm{Y},\ \mathrm{Z})])],\ [\mathrm{BY}\ \langle\mathrm{MANNER}\rangle])]$
	touch	47.8	$[\mathrm{BE}_{\mathrm{Loc}}\ (\mathrm{Y},[\mathrm{AT}_{\mathrm{Loc}}\ (\mathrm{Y},\ \mathrm{Z})],\ [\mathrm{BY}\ \langle\mathrm{MANNER}\rangle])]$
Motion	abandon	51.2	$[\mathrm{GO}_{\mathrm{Loc}}\ (\mathrm{Y},\ [\langle\mathrm{DIRECTION}\rangle_{\mathrm{Loc}}\ (\mathrm{Y},\ [\mathrm{AT}_{\mathrm{Loc}}\ (\mathrm{Y},\ \mathrm{Z})])])]$
	float	51.3.1	$[\mathrm{GO}_{\mathrm{Loc}}\ (\mathrm{Y},\ [\mathrm{BY}\ \langle\mathrm{MANNER}\rangle])]$
Placement	adorn	9.8	$[\mathrm{CAUSE}\ (\mathrm{X},$ $[\mathrm{GO}_{\mathrm{Ident}}\ (\mathrm{Y},$ $[\mathrm{TOWARD}_{\mathrm{Ident}}\ (\mathrm{Y},$ $[\mathrm{AT}_{\mathrm{Ident}}\ (\mathrm{Y},\ [\langle\mathrm{STATE}\rangle_{\mathrm{Ident}}\ ([\langle\mathrm{WITH}\rangle_{\mathrm{Poss}}\ (\mathrm{*HEAD*},$ $\mathrm{Z})])])])])])]$
	spill	9.5	$[\mathrm{CAUSE}\ (\mathrm{X},\ [\mathrm{GO}_{\mathrm{Loc}}\ (\mathrm{Y})],\ [\mathrm{BY}\ \langle\mathrm{MANNER}\rangle])]$

One piece of information that is not shown above is the mapping between the class/grid/lexeme input and the LCS representation. Using Levin's publicly available online index as a starting point, we have hand-constructed a database containing 191 LCS templates, i.e., one for verb class. These templates serve as the basis of instantiated output such as that of (3) above. A full entry in the database includes a list of possible lexemes, a thematic grid, and a LCS template:

(4) **Class 47.8:** adjoin, intersect, meet, touch, ...

[2]This is **not** to say that the generated LCS's for all translation equivalents are identical. Shortly, we will see a case where the LCS's produced by the program for two translation equivalents are annotated differently.

[3]Verbs not occurring in Levin's book are also assigned to classes using techniques described in (Dorr and Jones, 1996; Dorr, To appear).

Thematic Grid: `_th_loc`
LCS Template:
`(be loc (thing 2) (at loc (thing 2) (thing 11)) (!!-ingly 26))`

The semantic class and lexeme, together, uniquely identify the word sense, or LCS template, to which the verb refers.[4]

Note that the `!!` acts as a wildcard; it will be filled by the lexeme itself, and the resulting form is called a *constant*, i.e., the idiosyncratic part of the meaning that distinguishes among members of a verb class (in the spirit of (Grimshaw, 1993; Levin and Rappaport Hovav, To appear; Pinker, 1989; Talmy, 1985)). In addition, the output annotations (such as $*$) shown in (3) do not appear in the LCS template in (4). This is where the language-specific information is abstracted away from the language-independent meaning components. It is the *instantiated* lexical entry for *touch* that will have this type of information.

It is important to point out that, 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 our work 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, whereas, *hit* and *touch* do not. Table 2 shows three broad semantic categories and example verbs along with their associated LCS representations.[5]

In addition to defining a mapping between semantic classes and LCS meaning components, we have generated LCS templates for 26 additional classes that are *not* included in Levin's system. Several of these correspond to verbs that take sentential complements (such as *coerce*). The additional classes include, among others: *Coerce Verbs* (blackmail, bribe, cajole, ...); *Conspire Verbs* (legislate, protest, rebel, ...); *Exceed Verbs* (exceed, excel, outbid, ...); *Penetrate Verbs* (penetrate, permeate, pervade, ...).

4 Language-Specific Annotations

As described above, language-specific information such as the $*$-marker is not included in the templates of the LCS database. This information is added to the templates by processing the components of thematic grid specifications. In our on-going example (3), the thematic grid is `_th_loc`. From this it is determined that the *theme* and the *location*

[4] The words are assumed to have been disambiguated by techniques described in (Dorr and Jones, 1996; Dorr, To appear), during the assignment to semantic classes.

[5] LCS constituents that contain angle-bracketed "constants" correspond to the `!!` wildcard. For example, [BY ⟨MANNER⟩] corresponds to `!!-ingly`.

are both obligatory (in English) and should be annotated as such in the instantiated LCS. This language-specific correspondence is indicated through the use of the $*$-marker, which specifies the positions that are filled in by other LCS entries during sentence processing.

To illustrate this point, consider the structural divergence between the following English/Spanish equivalents; the lexical entries for the main verbs are included for comparison:

(5) (i) **Structural Divergence:**
E: John entered the house.
S: John entró a la casa.
'John entered into the house.'

(ii) **Lexical Entries:**
enter:
```
(go loc (* thing 2)
  (toward loc (thing 2) (in loc (thing 2) (* thing 6)))
  (enteringly 26))
```
entrar:
```
(go loc (* thing 2)
 ((* toward 5) loc (thing 2) (in loc (thing 2) (thing 6)))
   (enteringly 26))
```

The English sentence differs structurally from the Spanish in that the noun phrase *the house* corresponds to a prepositional phrase *a la casa*. The lexicon entries for *enter* and *entrar* both mean "X (= Thing 2) goes into location Y (= Thing 6)," but the distinction shown in (5)(i) is characterized by different positionings of the $*$-marker in (5)(ii). In the LCS entries, variable positions (designated by numbers, such as 2 and 6) are used in place of the ultimate fillers such as JOHN and HOUSE. A $*$-marked leaf node must be filled in at the leaf level and a $*$-marked non-leaf node must be filled in through unification at the internal node indicated by the $*$-marker. Hence, in the above example the LCS for *enter* guarantees that the second argument (position 6) is filled in at the leaf node, whereas the LCS for *entrar* requires this same argument to be filled through unification at the `(to loc ...)` node (position 5). In this latter case, the Spanish preposition *a* (i.e., 'to/into') fulfills this requirement since it has an LCS that matches at that level:
`(toward loc (thing 2) (in loc (thing 2) (thing 6)))`.

In addition to the $*$-marker, lexical entries contain other language-specific information: a `:collocations` slot for idiosyncratic prepositions and a `:var_spec` slot for information about variable positions (e.g., optionality of arguments). For brevity, we focus only on aspects of the acquisition process that affect the instantiation of the LCS template. The next section describes the construction of lexical entries in more detail.

Table 3: Thematic Roles Corresponding to Numbered Positions in the LCS: Modifiers

Arguments	
No./Role	**Description**
1/ag	Agent of CAUSE, LET, and CAUSE_EXCHANGE
2/th	Theme of GO_{Loc}, GO_{Poss}, GO_{Ident}, GO_{Circ}, BE_{Poss}, BE_{Ident}, BE_{Circ}
2/exp	Experiencer of GO_{Perc} and BE_{Perc}
3/src()	Path (from/from_away) before source
4/src	Source of Loc, Ident, Poss
5/goal()	Path (to/toward) preceding goal
6/goal	Goal of Loc, Ident, Poss
7/perc()	Perceived particle preceding perceived item
8/perc	Perceived item
9/pred	Identificational predicate
10/loc()	Locational particle preceding locative goal
11/loc	Locational predicate
12/poss	Possessional predicate
13/time	Temporal particle preceding time
14/time()	Time for Temp field

Modifiers	
No./Role	**Description**
15/mod-poss()	Possessional particle preceding possessed item
16/mod-poss	Possessed item modifier
17/ben()	Intentional particle preceding benefactive modifier
18/ben	Benefactive modifier
19/instr()	Instrumental particle preceding instrument modifier
20/instr	Instrument modifier
21/purp()	Intentional particle preceding purpose modifier
22/purp	Purpose modifier
23/mod-loc()	Locational particle preceding location modifier
24/mod-loc	Location modifier
25/manner()	Manner component
26/!!-ingly	Conflated manner component
27/prop	Event or state
28/mod-prop	Event or state
29/mod-pred()	Identificational particle preceding property modifier
30/mod-pred	Property modifier

5 Construction of Lexical Entries

As illustrated above, LCS templates use a Lisp-like representation. Consider *Fill Verbs* (9.8):[6]

(6)
```
(cause (thing 1)
   (go ident (thing 2)
      (toward ident (thing 2) (at ident (thing 2) (!!-ed 9))))
   (with poss (*head*) (thing 16)))
```

This list structure recursively associates logical heads with their arguments and modifiers. The logical head is represented as a primitive/field combination, e.g., $\mathrm{GO}_{\mathrm{Loc}}$ is represented as `(go loc ...)`. The arguments for CAUSE are `(thing 1)` and `(go ident ...)`. The substructure GO itself has two arguments `(thing 2)` and `(toward ident ...)`. Modifiers are non-argument positions in the LCS, e.g., the `(with poss ...)` substructure in the representation above (as in the sentence *She trimmed the tree with tinsel*).[7] Note that the `!!`-ed constant refers to a resulting state, e.g., `adorned` for the verb *adorn*.

The numbers in the representation are codes that map between LCS positions and their corresponding thematic roles. In the LCS framework, thematic roles provide semantic information about properties of the argument and modifier structures. We adopt the *thematic relations hypothesis* (Fillmore, 1968; Gruber, 1965) which states that the semantics of location and motion can be generalized to additional semantic fields.

Thematic roles have long been the focus of conceptual structure studies. In the work of (Jackendoff, 1983), thematic roles are associated with the notion of fields (Spatial, Temporal, Possessional, Circumstantial, and Existential) and have continued their prominence in more recent theories of lexical semantics (Dowty, 1991; Jackendoff, 1990). Subsequently, additional thematic roles have been proposed for new semantic fields (Dorr et al., 1993): Perceptual (for Events and States), Intentional (for modifiers such as purpose clauses), and Instrumental (for instrument modifiers).

Tables 3 shows the relation of thematic roles to components of the LCS representation for arguments and modifiers. These numbers enter into the construction of LCS entries: they correspond to argument positions in the LCS template (extracted using the class/grid/lexeme specification). Information is filled into the LCS template using these numbers, coupled with the thematic grid tag for the particular word being defined.

[6]Some of the 9.8 verbs are: *adorn, anoint, bandage, bathe, bestrew, bind, blanket, block, blot, festoon, fill, fleck, flood, frame, garland, garnish, imbue, mottle, ornament, pad, pave, plate, plug, pollute, replenish, stud, suffuse, surround, swaddle, swathe, taint, tile, trim, veil.*

[7]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.

5.1 Fundamentals

LEXICALL is designed to locate the appropriate LCS template (out of the 191 hand-constructed database entries) using the class/grid pairing as an index, and to then determine the language-specific annotations to instantiate for that template. Although it is desirable to generate the most representative LCS definition for a lexical item, this is not always possible. A verb class usually contains template variations for the verbs they cover, i.e., subclasses. These are relatively general cases—verbs in a class show similar properties, but the class is further refined by certain template variations.

Thus, for each verb to be defined, LEXICALL uses the class/grid pairing to extract the smallest possible set of LCS templates that comply with the class/grid pairing. It is also possible to index into the database by the grid only. The semantic class is considered an optional argument to the acquisition procedure, and generally more than one LCS definition will be returned in such cases. The grid information cannot be omitted, however.

Once a set of LCS templates is selected, the next step is to place $*$-markers in the appropriate positions. To do this, the program must take into account structural divergence possibilities (e.g., the case given in example (5)(i) above). The default position of the $*$-marker is the left-most occurrence of the LCS node corresponding to a particular thematic role. However, if a preposition occurs in the grid, the $*$-marker may be placed differently. In such a case, a primitive representation (e.g., `(to loc (at loc))`) is extracted using a set of predefined mappings (to be described in Section 5.2, case III). If this representation corresponds to a subcomponent of the LCS template, the program recognizes this as a match against the grid, and the $*$-marker is placed in the template at the level where this match occurs (as in the entry for *entrar* given in (5)(ii) above).

If a preposition occurs in the grid but there is no matching primitive representation, the preposition is considered to be a collocation, and it is placed in the `collocations` slot of the lexical entry—which indicates that the LCS already covers the semantics of the verb and the preposition is an idiosyncratic variation (as in *learn about*, *know of*, etc.).

If a preposition is required but it is not specified (i.e., empty parentheses ()), then the $*$-marker is positioned at the level dominating the node that corresponds to that role—which indicates that several different prepositions might apply (as in *put on*, *put under*, *put through*, etc.).

5.2 Processing thematic grids

The input to LEXICALL is a class/grid/lexeme specification, where each piece of information is separated by a hash sign (`#`):

`<class>#<grid>#<lexeme>#<other semantic information>` For example, the input specification for the verb *replant* (a word not classified by Levin) is:

```
9.7#_ag_th,mod-poss(with)#replant#!!-ed =
planted (manner = again)
```

This input indicates that the class assigned to *replant* is 9.7 (Levin's Spray/Load verbs) and its grid has an obligatory agent (`ag`), theme (`th`), and an optional possessional modifier with preposition *with* (`mod-poss(with)`). By default, arguments are obligatory and modifiers are optional. If a role in the thematic grid overrides this default, the information is recorded in the `var_spec` slot of the lexical item.

The information following the final `#` is optional; this information was previously hand-added to the assigned thematic grids. In the current example, the `!!-ed` designates the form of the constant *planted* which, in this case, is a morphological variant of the lexeme *replant*.[8] Also, the manner *again* is specified as an additional semantic component.

Returning to our example (5) above, the verbs *entrar* and *entrar* have the following class/grid/lexeme specifications:

(7) **enter:** `51.1#_th,goal#enter#`
entrar: `51.1#_th,goal(a)#enter#`

LEXICALL produces the English and Spanish LCS output shown in (5)(ii) for these two specifications.

For presentational purposes, the remainder of this section uses English examples. However, as the above examples illustrate, the class/grid/lexeme specification carries over to other languages as well. In fact, we have used the same acquisition program, without modification, for building our Spanish and Arabic LCS-based lexicons, each of size comparable to our English LCS-based lexicon.

There are a number of different cases for which the class/grid/lexeme specification needs to be processed for the construction of lexical entries. We will look at four cases in detail.

I. Thematic Roles without Prepositions

(8) **Example:** The flower decorated the room.
Input: `9.8#_mod-poss_th#decorate#`
Template:
```
(be ident (thing 2) (at ident (thing 2)
                    (!!-ed 9))
                    (with poss (*head*) (thing 16)))
```

[8]The constant takes one of several forms, including: `!!-ingly` for a manner, `!!-er` for an instrument, and `!!-ed` for resulting states. If this information has not been hand-added to the class/grid/lexeme specification (as is the case with most of the verbs), a default morphological process produces the appropriate form from the lexeme. Note that, in some cases, the morphological process might produce unexpected surface forms (e.g., *hoer* instead of *hoe*). Because we take the constant to be a *label* for semantic meaning—having a different status from other components of meaning (e.g., GO, BE, etc.)—the precise form of the constant is inconsequential; thus, a close approximation is sufficient for our purposes.

This case involves thematic roles that do not require prepositions for any of the argument or modifier variables. Two thematic roles, `th` and `mod-poss`, are specified for the above sense of the English verb *decorate*. Table 3 indicates that the thematic code numbers are 2 and 16, respectively. The following output shows the result of positioning the ∗-markers and the constant *decorated* in the LCS template:

(9) **Output:**
```
(be ident (* thing 2) (at ident (thing 2)
                       (decorated 9))
                       (with poss (*head*) (* thing 16)))
```

Duplicate occurrences of a node are taken to be coindexed; thus, only one such node needs to be ∗-marked. Our convention is to mark the left-most occurrence (e.g., the first `(thing 2)` node above).

II. Thematic Roles with Unspecified Prepositions

(10) **Example:** We parked the car near the store.
We parked the car in the garage.

Input: `9.1#_ag_th_goal()#park#`

Template:
```
(cause (thing 1)
       (go loc (thing 2) (toward loc (thing 2)
          ([at] loc (thing 2) (thing 6))))
       (!!-ingly 26))
```

The input for this example indicates that there is necessarily a goal headed by a preposition, although the preposition is not given explicitly. The thematic roles are `ag`, `th`, and `goal()` with the corresponding code numbers 1, 2, and 6, as given in Table 3. The resulting output is the following:

(11) **Output:**
```
(cause (* thing 1)
       (go loc (* thing 2) ((* toward 5) loc (thing 2)
          ([at] loc (thing 2) (thing 6))))
       (parkingly 26))
```

The variable positions for `ag` and `th` are marked just as in the previous case, whereas `goal()` requires a different treatment. When there is no preposition specified for the thematic roles with required prepositions, the ∗-marker is associated with a LCS node dominating the position corresponding to the relevant thematic role (i.e., `(toward ...)` in the example above). The ability to underspecify the preposition allows for a number of surface-sentence variants such as *park in*, *park on*, *park under*, etc. (In such cases, the bracketed primitive `[at]` node will be converted into an unbracketed positional primitive—e.g., `in`, `on`, or `under`—during analysis of the surface sentence.)

III. Thematic roles with Specified Prepositions

(12) **Example:** We decorated the room with flowers.
Input: `9.8#_ag_th,mod-poss(with)#decorate#`
Template:
```
(cause (thing 1)
       (go ident (thing 2) (toward ident (thing 2)
         (at ident (thing 2) (!!-ed 9))))
       (with poss (*head*) (thing 16)))
```

Specified prepositions can occur both for argument and modifier thematic roles. This example demonstrates the process for a modifier role, however the same also applies to the argument roles. Here, the `mod-poss` role requires the preposition *with* in the modifier position. Thus, the output of LEXICALL is:

(13) **Output:**
```
(cause (* thing 1)
       (go ident (* thing 2) (toward ident (thing 2)
         (at ident (thing 2) (decorated 9))))
       ((* with 15) poss (*head*) (thing 16)))
```

In order to determine the position of the ∗-marker for a thematic role with a required preposition, LEXICALL consults a set of predefined mappings between prepositions (or postpositions, in a language like Korean) and their corresponding primitive representations.[9] In the current case, the preposition *with* is mapped to the following primitive representation: `(with poss)`. Since this matches a sub-component of the LCS template, the program recognizes this as a match against the grid, and the ∗-marker is placed in the template at the level of `with`.

IV. Thematic Roles with Extra information

(14) **Example:** They afforested the farm.
Input:
```
9.8#_ag_goal#afforest#!!-ingly =
        afforestingly (th = forest)
```
Template:
```
(cause (thing 1)
       (go loc (thing 2) (toward loc (thing 2)
         (at loc (thing 2) (thing 6))))
       (with loc (*head*) (thing 16)))
       (!!-ingly 26))
```

[9] We have defined approximately 100 such mappings per language. For example, the mapping produces the following primitive representations for the English word *to*: `(to loc (at loc))`, `(to poss (at poss))`, `(to temp (at temp))`, `(toward loc (at loc))`, `(toward poss (at poss))`. We have similar mappings defined in Spanish and Arabic. For example, the following primitive representations are produced for the Spanish word *a*: `(at loc)`, `(to loc (at loc))`, `(to poss (at poss))`, `(toward loc (at loc))`.

Recall that the input potentially has hand-coded information following the final `#`. In such cases, the specification includes a constant (as in the manner `afforestingly` above) and sometimes extra information about LCS arguments. In the example above, the specification indicates that the verb *afforest* has an implicit theme in its meaning, i.e., *forest*. Thus, instead of annotating the theme (designated by the code 2) with a $*$-marker, this position is directly replaced by the primitive `forest`:

(15) **Output:**
```
(cause (* thing 1)
       (go loc (forest 2) (toward loc (forest 2)
          (at loc (forest 2) (* thing 6))))
       (with loc (*head*) (thing 16)))
       (afforestingly 26))
```

6 Limitations and Conclusions

We have described techniques for automatic construction of dictionaries for use in large-scale FLT. The dictionaries are based on a language-independent representation called *lexical conceptual structure* (LCS). Significant enhancements to LCS-based tutoring could be achieved by combining this representation with a mechanism for handling issues related to discourse and pragmatics. For example, although the LCS processor is capable of determining that the phrase *in the trash* partially matches the answer to *Where did John put the book?*, a pragmatic component would be required to determine that this answer is (perhaps) more appropriate than the full answer, *He put the book in the trash*. Representing conversational context and dynamic context updating (Traum et al., 1996; Haller, 1996; DiEugenio and Webber, 1996) would provide a framework for this type of response "relaxation." Along these same lines, a pragmatic component could provide a mechanism for determining that certain fully matched responses (e.g., *John hurled the book into the trash*) are not as "realistic sounding" as partially matched alternatives.

Initially, LEXICALL was designed to support the development of LCS's for English only; however, the same techniques can be used for multilingual acquisition. As the lexicon coverage for other languages expands, it is expected that our acquisition techniques will help further in the cross-linguistic investigation of the relationship between Levin's verb classes and the basic meaning components in the LCS representation. In addition, it is expected that verbs in the same Levin class may have finer distinctions than what we have specified in the current LCS templates.

We view the importation of LCS's from the English LCS database into Arabic and Spanish as a first approximation to the development of complete lexicons for these languages. The results have been hand-checked by native speakers using the class/grid/lexeme format (which is much easier to check than the fully expanded LCS's). The lexical verification process took only two weeks by the native speakers. We 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.

One important benefit of using the Levin classification as the basis of our program 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 191 classes. We see our approach as a first step toward compression of lexical entries in that it allows lexicons to be stored in terms of the more condensed class/grid/lexeme specifications; these can expanded online, as needed, during sentence processing in the NLP application.

We conclude that, while human intervention is necessary for the acquisition of class/grid information, this intervention is virtually eliminated from the LCS construction process because of our provision of a mapping between semantic classes and primitive meaning components.

Acknowledgements

I would like to thank Jungshin Park and Mine Ulku Sencan for their aid in the development of certain components of the LEXICALL program. This work was supported, in part, by 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.

References

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

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

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

Ann Copestake, 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.

Barbara DiEugenio and Bonnie Lynn Webber. 1996. Pragmatic Overloading in Natural Language Instructions. *International Journal of Expert Systems*, 9(1):53–84.

Bonnie J. Dorr and Douglas Jones. 1996. 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.

Bonnie J. Dorr, 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.

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

Bonnie J. Dorr. To appear. Large-Scale Dictionary Construction for Foreign Language Tutoring and Interlingual Machine Translation. *Machine Translation*, 12(1).

David Dowty. 1991. The Effects of Aspectual Class on the Temporal Structure of Discourse: Semantics or Pragmatics? *Language*, 67:547–619.

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

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

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

Jane Grimshaw. 1993. Semantic Structure and Semantic Content in Lexical Representation. unpublished ms., Rutgers University, New Brunswick, NJ.

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

Ken Hale and Samuel J. Keyser. 1993. 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*. MIT Press, Cambridge, MA.

Susan Haller. 1996. Planning Text About Plans Interactively. *International Journal of Expert Systems*, 9(1):85–112.

Melissa Holland. 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.

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

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

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

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

Beth Levin 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.

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

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

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

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

Michelle Sams. 1993. An Intelligent Foreign Language Tutor Incorporating Natural Language Processing. In *Proceedings of Conference on Intelligent Computer-Aided Training and Virtual Environment Technology*, NASA: Houston, TX.

Leonard Talmy. 1985. Lexicalization Patterns: Semantic Structure in Lexical Forms. In T. Shopen, editor, *Language Typology and Syntactic Description 3: Grammatical Categories and the Lexicon*, pages 57–149. University Press, Cambridge, England.

David R. Traum, Lenhart K. Schubert, Nathaniel G. Martin, Chung Hee Hwang, Peter Heeman, George Ferguson, James Allen, Massimo Poesio, and Marc Light. 1996. Knowledge Representation in the TRAINS-93 Conversation System. *International Journal of Expert Systems*, 9(1):173–223.

Amy Weinberg, 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.

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