Structural properties of Lexical Systems: Monolingual and Multilingual Perspectives

Alain Polguère
OLST—Département de linguistique et de traduction
Université de Montréal
C.P. 6128, succ. Centre-ville, Montréal (Québec) H3C 3J7 Canada
alain.polguere@umontreal.ca

Abstract

We introduce a new type of lexical structure called *lexical system*, an interoperable model that can feed both monolingual and multilingual language resources. We begin with a formal characterization of lexical systems as "pure" directed graphs, solely made up of nodes corresponding to lexical entities and links. To illustrate our approach, we present data borrowed from a lexical system that has been generated from the French DiCo database. We later explain how the compilation of the original dictionary-like database into a net-like one has been made possible. Finally, we discuss the potential of the proposed lexical structure for designing multilingual lexical resources.

1 Introduction

The aim of this paper is to introduce, justify and exemplify a new type of structure for lexical resources called *lexical systems*. Although lexical systems are basically monolingual entities, we believe they are particularly well-suited for the implementation of interlingual connections.

Our demonstration of the value of lexical systems is centered around an experiment of lexical system generation that was performed using data tables extracted from the DiCo database of French paradigmatic and syntagmatic lexical links. This experiment has allowed us to produce a lexical system that is a richer structure than the original database it has been derived from.

In section 2, we characterize two main families of lexical databases presently available: dictionary-like *vs.* net-like lexical databases; we then proceed with describing the specific structure of lexical systems. Section 3 illustrates the functioning of lexical systems with data borrowed from the French DiCo database; this will show that lexical systems—that are basically net-like—are interoperable structures in respect to the information they can easily encode and the wide range of applications for which they can function as lexical resources. Section 4 describes how the generation of a lexical system from the French DiCo database has been implemented. Finally, in section 5, we address the problem of using lexical systems for feeding multilingual databases.

2 Structure of lexical systems

Lexical systems as formal models of natural language lexica are very much related to the "-*Net*" generation of lexical databases, whose most well-known representatives are undoubtedly WordNet (Fellbaum, 1998) and FrameNet (Baker *et al.*, 2003). However, lexical systems possess some very specific characteristics that clearly distinguish them from other lexicographic structures. We will first characterize the two main current approaches to the structuring of lexical models and then present lexical systems relative to them.

2.1 Dictionary- vs. net-like lexical databases

Dictionary-like databases as texts

The most straightforward way of building lexical databases is to use standard dictionaries (i.e. books) and turn them into electronic entities. It is the approach taken by most publishing companies (e.g. American Heritage (2000)), with various degrees of sophistication. Resulting products

Proceedings of the Workshop on Multilingual Language Resources and Interoperability, pages 50–59,
Sydney, July 2006.

can be termed *dictionary-like databases*. They are mainly characterized by two features.

- They are made up of word (word sense) descriptions, called *dictionary entries*.
- Dictionary entries can be seen as "texts," in the most general sense.

Consequently, dictionary-like databases are before all huge texts, consisting of a collection of much smaller texts (i.e. entries).

It seems natural to consider electronic versions of standard dictionaries as texts. However, formal lexical databases such as the multilingual XML-based JMDict (Breen, 2004) are also textual in nature. There are collections of entries, each entry consisting of a structured text that "tells us something" about a word. Even databases encoding relational models of the lexicon can be 100% textual, and therefore dictionary-like. Such is the case of the French DiCo database (Polguère, 2000), that we have used for compiling our lexical system. As we will see later, the original DiCo database is nothing but a collection of lexicographic records, each record being subdivided into fields that are basically small texts. Although the DiCo is built within the framework of Explanatory Combinatorial Lexicology (Mel'čuk *et al.*, 1995) and concentrates on the description of lexical links, it is clearly not designed as a "*-Net*" database, in the sense of WordNet or FrameNet.

Net-like databases as graphs

Most lexical models, even standard dictionaries, are relational in nature. For instance, all dictionaries define words in terms of other words, use pointers such as 'Synonym' and 'Antonym.' However, their structure does not reflect their relational nature. The situation is totally different with true net-like databases. They can be characterized as follows.

- They are graphs—huge sets of connected entities—rather than collections of small texts (entries).
- They are not necessarily centered around words, or word senses. They use as nodes a potentially heterogeneous set of lexical or, more generally, linguistic entities.

Net-like databases are, for many, the most suitable knowledge structures for modeling lexica. Nevertheless, databases such as WordNet pose one major problem: they are inherently structured according to a couple of hierarchizing and/or classifying principles. WordNet, for instance, is semantically-oriented and imposes a hierarchical organization of lexical entities based, first of all, on two specific semantic relations: synonymy—through the grouping of lexical meanings within *synsets*—and hypernymy. Additionally, the part of speech classification of lexical units creates a strict partition of the database: WordNet is made up of four separate synset hierarchies (for nouns, verbs, adjectives and adverbs). We do not believe lexical models should be designed following a few rigid principles that impose a hierarchization or classification of data. Such structuring is of course extremely useful, even necessary, but should be projected "on demand" onto lexical models. Furthermore, there should not be a predefined, finite set of potential structuring principles; data structures should welcome any of them, and this is precisely one of the main characteristics of lexical systems, that will be presented shortly (section 2.2).

Texts *vs.* graphs: pros and cons

It is essential to stress the fact that any dictionary-like database can be turned into a net-like database and vice versa. Of course, dictionary-like databases that rely on relational models are more compatible with graph encoding. However, there are always relational data in dictionaries, and such data can be extracted and "reformatted" in the form of nodes and connecting links.

The important issue is therefore not one of exclusive choice between the two types of structures; it concerns what each structure is better at. In our opinion, the specialization of each type of structure is as follows.

Dictionary-like structures are tools for editing (writing) and consulting lexical information. Linguistic intuition of lexicographers or users of lexical models performs best on texts. Both lexicographers and users need to be able to see the whole picture about words, and need the entry format at a certain stage—although other ways of displaying lexical information, such as tables, are extremely useful too![1]

Net-like structures are tools for implementing dynamic aspects of lexica: wading through lexical knowledge, adding to it, revising it or infer-

[1] It is no coincidence if WordNet so-called *lexicographer files* give a textual perspective on lexical items that is quite dictionary-like. The unit of description is the synset, however, and not the lexical unit. (See WordNet on-line documentation on lexicographer files.)

ring information from it. Consequently, net-like databases are believed by some (and we share this opinion) to have some form of cognitive validity. They are compatible with observations made, for instance, in Aitchison (2003) on the network nature of the mental lexicon. Last but not least, net-like databases can more easily integrate other lexical structures or be integrated by them.

In conclusion, although both forms of structures are compatible at a certain level and have their own advantages in specific contexts of use, we are particularly interested by the fact that net-like databases are more prone to live an "organic life" in terms of evolution (addition, subtraction, replacement) and interaction with other data structures (connection with models of other languages, with grammars, etc.).

2.2 Lexical systems: a new type of net-like lexical databases

As mentioned above, most net-like lexical databases seem to focus on the description of just a few properties of natural language lexica (quasi-synonymy, hypernymic organization of word senses, predicative structures and their syntactic expression, etc.). Consequently, developers of these databases often have to gradually "stretch" their models in order to add the description of new types of phenomena, that were not of primary concern at the onset. It is legitimate to expect that such graft of new components will leave scars on the initial design of lexical models.

The lexical structures we propose, lexical systems (hereafter *LS*), do not pose this type of problem for two reasons.

First, they are not oriented towards the modeling of just a few specific lexical phenomena, but originate from a global vision of the lexicon as central component of linguistic knowledge.

Second, they have a very simple, flat organization, that does not impose any hierarchical or classifying structure on the lexicon. Let us explain how it works.

The design of any given LS has to follow four basic principles, that cannot be tampered with: LSs are 1) pure directed graphs, 2) non-hierarchical, 3) heterogeneous and 4) equipped for modeling fuzziness of lexical knowledge. We will briefly examine each of these principles.

Pure directed graph. An LS is a directed graph, and just that. This means that, from a formal point of view, it is **uniquely** made up of nodes and oriented links connecting these nodes.

Non hierarchical. An LS is a non-hierarchical structure, although it can contain sets of nodes that are hierarchically connected. For instance, we will see later that the DiCo LS contains nodes that correspond to a hierarchically organized set of semantic labels. The hierarchy of DiCo semantic labels can be used to project a structured perspective on the LS; but the LS itself is by no means organized according to one or more specific hierarchies.

Heterogeneous. An LS is a potentially heterogeneous collection of nodes. Three main families of nodes can be found:

- genuine lexical entities such as lexemes, idioms, wordforms, etc.;
- quasi-lexical entities, such as collocations, lexical functions,[2] free expressions worth storing in the lexicon (e.g. "canned" linguistic examples), etc.;
- lexico-grammatical entities, such as syntactic patterns of expression of semantic actants, grammatical features, etc.

Prototypical LS nodes are first of all lexical entities, but we have to expect LSs to contain as nodes entities that do not strictly belong to the lexicon: they can belong to the interface between the lexicon and the grammar of the language. Such is the case of subcategorization frames, called *government patterns* in Explanatory Combinatorial Lexicology. As rules specifying patterns of syntactic structures, they belong to the grammar of the language. However, as preassembled constructs on which lexemes "sit" in sentences, they are clearly closer to the lexical realm of the language than rules for building passive sentences or handling agreements, for instance.

With fuzziness. Each component of an LS, whether node or link, carries a trust value, i.e. a measure of its validity. Clearly, there are many ways of attributing and handling trust values in order to implement fuzziness in knowledge structures. For instance, in our experiments with the DiCo LS, we have adopted a simplistic approach, that was satisfactory for our present needs but should become more elaborate as we proceed with developing and using LSs. In our present implementation, we make use of only three possible trust values: "`1`" means that as far as we can tell—i.e. trusting what is explicitly asserted in the DiCo—the information is correct; "`0.5`" means

[2] On collocations and lexical functions, see section 3 below.

that the corresponding information is the result of an inference made from the input data and was not explicitly asserted by lexicographers; "0" means that the information ought to be incorrect—for instance, in case we identified a bogus lexical pointer in data imported from the DiCo.

Fuzziness encoding is an essential feature of LSs, as structures on which inference can take place or as structures that are, at least partially, inferred from others (in case of generation of LSs from existing lexical databases). Of course, any trust value is not absolute. "1" does not mean the information is valid no matter what, and "0" that it is necessarily false. Information in LSs, and the rating of this information, is no more absolute than any information that may be stored in someone's mental lexicon. However, if we want to compute on LSs' content, it is essential to be able to distinguish between data we have all reasons to believe to be true and data we have all reasons to believe to be false. As a matter of fact, this feature of LSs has helped us in two ways while compiling the DiCo LS: (i) we were able to infer new descriptions from data contained in the original DiCo while keeping track of the inferred nature of this new information (that ought to be validated); (ii) we kept record of incoherences found in the DiCo by attributing a trust value of 0 to the corresponding elements in the LS.

It is now high time to give concrete examples of LS data. But before we proceed, let us emphasize the fact that no other formal devices than those that have just been introduced are allowed in LSs. Anything else we may want to add must be relevant to other components of the linguistic model, to the grammar for instance. Notice, however, that we do not exclude the need to add a measure of the relative "weight" of nodes and links. This measure, different from the trust value, would reflect the degree of activation of each LS element. For instance, the DiCo entry for DÉFAITE 'defeat' lists quite a few support verbs that take this noun as complement, among which CONNAÎTRE 'to know' and SUBIR 'to suffer.' Weight values could indicate that the former verb is much less commonly used than the second in this context. Another advantage of weight is that it could help optimize navigation through the LS graph, when several paths can be taken.

3 Examples borrowed from the DiCo LS

The DiCo is a French lexical database that focuses on the modeling of paradigmatic and syntagmatic lexical links controlled by lexical units. Paradigmatic links correspond to so-called *semantic derivations* (synonymy, antonymy, nominalization, verbalization, names for actants or typical circonstants, etc.). Syntagmatic links corresponds to collocations controlled by lexical units (intensifiers, support verbs, etc.). These lexical properties are encoded by means of a system of metalexical entities known as *lexical functions*. (For a presentation of the system of lexical functions, see Mel'čuk (1996) and Kahane and Polguère (2001).) Although it does not contain actual definitions, the DiCo partially describes the semantic content of each lexical unit with two formal tools: (i) a semantic label, that corresponds to the genus (core component) of the lexical unit's definition and (ii) a "propositional formula," which states the predicative nature of the unit (non-predicative meaning or predicate with one, two or more arguments). Each entry also gives the government pattern (roughly, the subcategorization frame) of the unit and lists idioms (phrasal lexical units) that contain the unit under description. Finally, each entry contains a set of examples retrieved from corpora or the Internet. As one can see, the DiCo covers a fairly large range of lexical properties; for more information on the DiCo, one can refer to Polguère (2000) and Lareau (2002).

Presently, the DiCo is developed as a FileMaker® database. Each DiCo entry corresponds to a record in the database, and the core of each record is the field that contains lexical function links controlled by the *headword* (i.e. the lexical unit described in the entry). Data in (1) below is one item in the lexical function field of the DiCo record for Fr. RANCUNE ('resentment'):

(1)
```
/*[X] éprouver ~*/
{Oper12} avoir, éprouver, nourrir,
         ressentir
         [ART ~ Prép-envers N=Y]
```

We isolate five different types of LS entities in the above example:

- The expression between curly brackets Oper12 is the name of a lexical function denoting a type of support verbs.[3]
- {Oper12} as a whole denotes Oper12(RANCUNE), the application of

[3] More precisely, Oper12 denotes support verbs that take the 1st actant of the headword as subject, the headword itself as 1st complement and the 2nd actant of the headword as 2nd complement; for instance: *X **feels**/**has** resentment for Y.*

the `Oper12` lexical function to its argument (the headword of the entry).

- The preceding formula—between the two /*…*/ symbols—is a gloss for `Oper12`(RANCUNE). This metalinguistic encoding of the content of the lexical function application is for the benefit of users who do not master the system of lexical functions.
- Following the name of the lexical function is the list of values of the lexical function application, each of which is a specific lexical entity. In this case, they are all collocates of the headword, due to the syntagmatic nature of `Oper12`.
- Finally, the expression between square brackets is the description of the syntactic structure controlled by the collocates. It corresponds to a special case of lexico-grammatical entities mentioned earlier in section 2.2. These entities have not been processed yet in our LS and they will be ignored in the discussion below.

Data in (1) corresponds to a very small subgraph in the generated LS, which is visualized in Figure 1 below. Notice that graphical representations we used here have been automatically generated in GraphML format from the LS and then displayed with the yEd graph editor/viewer.

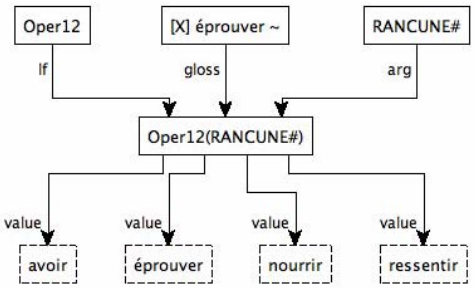

Figure 1. LS interpretation of (1)

This graph shows how DiCo data given in (1) have been modeled in terms of lexical entities and links. We see that lexical function applications are lexical entities: something to be communicated, that is pointing to actual means of expressing it. The argument (`arg` link) of the lexical function application, the lexical unit RANCUNE, is of course also a lexical entity (although of a different nature). The same holds for the values (`value` links). None of these values, however, has been diagnosed as possessing a corresponding entry in the DiCo. Consequently, the compilation process has given them the (temporary) status of simple wordforms, with a trust value of `0.5`, visualized here by boxes with hashed borders. (Continuous lines for links or boxes indicate a trust value of `1`.) Ultimately, it will be the task of lexicographers to add to the DiCo entries for the corresponding senses of AVOIR, ÉPROUVER, NOURRIR and RESSENTIR.

One may be surprised to see lexical functions (such as `Oper1`) appear as lexical entities in our LS, because of their very "abstract" nature. Two facts justify this approach. First, lexical units too are rather abstract entities. While wordforms *horse* and *horses* could be considered as more "concrete," their grouping under a label *HORSE lexical unit* is not a trivial abstraction. Second, lexical functions are not only descriptive tools in Explanatory Combinatorial Lexicology. They are also conceptualized as generalization of lexical units that play an important role text production, in general rules of paraphrase for instance.

This first illustration demonstrates how the LS version of the DiCo reflects its true relational nature, contrary to its original dictionary-like format as a FileMaker database. It also shows how varied lexical entities can be and how trust values can help keep track of the distinction between what has been explicitly stated by lexicographers and what can be inferred from what they stated.

The next illustration will build on the first one and show how so-called *non-standard lexical functions* are integrated into the LS. Until now, we have been referring only to standard lexical functions, i.e. lexical functions that belong to the small universal core of lexical relations identified in Explanatory Combinatorial Lexicology (or, more generally, in Meaning-Text theory). However, all paradigmatic and syntagmatic links are not necessarily standard. Here is an illustration, borrowed from the DiCo entry for CHAT 'cat'.

(2) `{Ce qu'on dit pour appeler ~} « Minet ! », « Minou ! », « Petit ! »`

Here, a totally non-standard lexical function `Ce qu'on dit pour appeler ~` 'What one says to call ~ [= a cat]' has been used to connect the headword CHAT to expressions such as *Minou !* 'Kitty kitty!' As one can see, no gloss has been introduced, because non-standard lexical functions are already explicit, non-formal encoding of lexical relations. The LS interpretation of (2) is therefore a simpler structure than the

one used in our previous illustration, as shown in Figure 2.

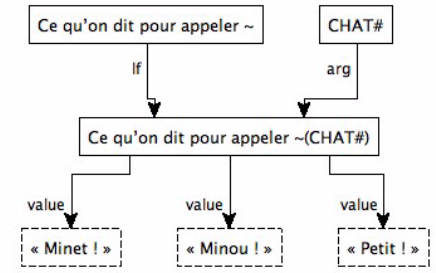

Figure 2. LS interpretation of (2)

Our last illustration will show how it is possible to project a hierarchical structuring on the DiCo LS when, **and only when**, it is needed.

The hierarchy of semantic labels used to semantically characterize lexical units in the DiCo has been compiled into the DiCo LS together with the lexical database proper. Each semantic label is connected to its more generic label or labels (as this hierarchy allows for multiple inheritance) with an `is_a` link. Additionally, it is connected to the lexical units it labels by `label` links. It is thus possible to simply pull the hierarchy of semantic labels out of the LS and it will "fish out" all lexical units of the LS, hierarchically organized through hypernymy. Notice that this is different from extracting from the DiCo all lexical units that possess a specific semantic label: we extract all units **whose semantic label belongs to a given subhierarchy** in the system of semantic labels. Figure 3 below is the graphical result of pulling the `accessoire` ('accessory') subhierarchy.

To avoid using labels on links, we have programmed the generation of this class of GraphML structures with links encoded as follows: `is_a` links (between semantic labels) appear as thick continuous arrows and `label` links (between semantic labels and lexical units they label) as thin dotted arrows.

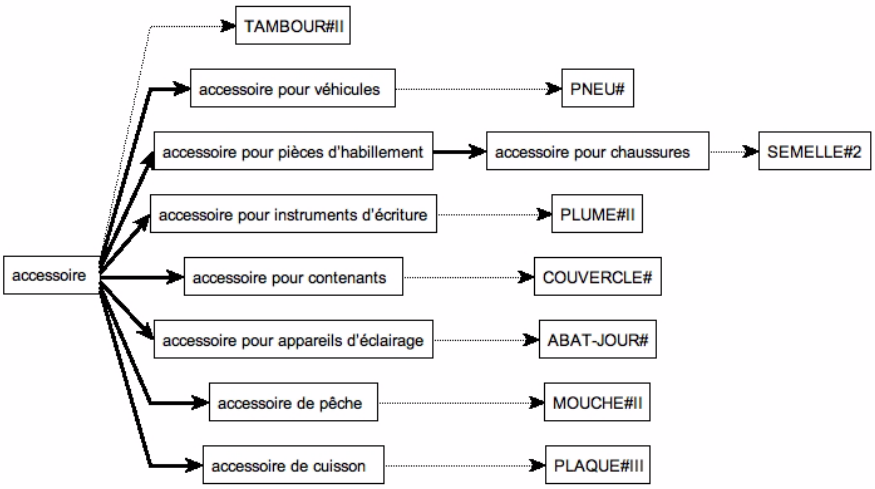

Figure 3. The `accessoire` ('accessory') semantic subhierarchy in the DiCo LS

The "beauty" of LSs' structuring does not lie in the fact that it allows us to automatically generate fancy graphical representations. Such representations are just a convenient way to make explicit the internal structure of LSs. What really interests us is what can be done with LSs once we consider them from a **functional** perspective.

The main functional advantage of LSs lies in the fact that these structures are both cannibal and prone to be cannibalized. Let us explain the two facets of this somehow gruesome metaphor.

First, directed graphs are powerful structures that can encode virtually any kind of information and are particularly suited for lexical knowledge. If one believes that a lexicon is before all a relational entity, we can postulate that all information present in any form of dictionary and database can eventually be compiled into LS structures. The experiment we did in compiling the DiCo (see details in section 4) demonstrates well enough this property of LS structures.

Second, because of their extreme simplicity, LS structures can conversely always be "digested" by other, more specific types of structures, such as XML versions of dictionary- or net-like databases. For instance, we have regenerated from our LS a DiCo in HTML format, with hyperlinks for entry cross-references and color-coding for trust values of linguistic information. Interestingly, this HTML by-product of the LS

contains entries that do not exist in the original DiCo. They are produced for each value of lexical function applications that does not correspond to an entry in the DiCo. The content of these entries is made up of "inverse" lexical function relations: pointers to lexical function applications for which the lexical entity is a value. These new entries can be seen as rough drafts, that can be used by lexicographers to write new entries. We will provide more details of this at the end of the next section.

4 Compiling the DiCo (dictionary-like) database into a lexical system

The DiCo is presently available both in FileMaker format and as SQL tables, accessible through the DiCouèbe interface.[4] It is these tables that are used as input for the generation of LSs.[5] They present the advantage of being the result of an extensive processing of the DiCo that splits its content into elementary pieces of lexicographic information (Steinlin *et al.*, 2005). It is therefore quite easy to analyze them further in order to perform a restructuring in terms of LS modeling.

The task of inferring new information, information that is not explicitly encoded in the DiCo, is the delicate part of the compilation process, due to the richness of the database. Until now, we have only implemented a small subset of all inferences that can be made. For instance, we have inferred individual lexemes from idioms that appear inside DiCo records (COUP DE SOLEIL 'sunburn' entails the probable existence of the three lexemes COUP, DE and SOLEIL). We have also distinguished lexical entities that are actual lexical units from their signifiers (linguistic forms). Signifiers, which do not have to be associated with one specific meaning, play an important role when it comes to wading through an LS (for instance, when we want to separate word access through form and through meaning).

We cannot give here all details of the compilation process. Suffice it to say that, at the present stage, some important information contained in the DiCo is not processed yet. For instance, we have not implemented the compilation of government patterns and lexicographic examples. On the other hand, all lexical function applications and the semantic labeling of lexical units are properly handled. Recall that we import together with the DiCo a hierarchy of semantic labels used by the DiCo lexicographers, which allows us to establish hypernymic links between lexical units, as shown in Figure 3 above.[6] Codewise, the DiCo LS is just a flat Prolog database with clauses for only two predicates:

```
entity( <Numerical ID>, <Name>,
        <Type>, <Trust> )
link( <Numerical ID>, <Source ID>,
      <Target ID>, <Type>, <Trust> )
```

Here are some statistics on the content of the DiCo LS at the time of writing.

Nodes : **37,808**

780 semantic labels; **1,301** vocables (= entries in the "LS wordlist"); **1,690** lexical units (= senses of vocables); **6,464** wordforms; **2,268** non lexicalized expressions; **7,389** monolexical signifiers; **948** multilexical signifiers; **3,443** lexical functions; **9,417** lexical function applications; **4,108** glosses of lexical function applications

Links : **61,714**

871 "is_a," between semantic labels; **775** "sem_label," between sem. labels and lexical units; **1,690** "sense," between vocables and lexical units corresponding to specific senses; **2,991** "basic_form," between mono- or multilexical signifiers and vocables or lexical units; **6,464** "signifier," between wordforms and monolexical signifiers; **4,135** "used_in," between monolexical signifiers and multiliexical signifiers; **9,417** "lf," between lexical functions and their application; **6,064** "gloss," between lex. func. appl. and their gloss; **9,417** "arg," between lex. func. appl. and their argument; **19,890** "value," between lex. func. appl. and each of the value elements they return

Let us make a few comments on these numbers in order to illustrate how the generation of the LS from the original DiCo database works.

The FileMaker (or SQL) DiCo database that has been used contained only 775 lexical unit records (word senses). This is reflected in statistics by the number of `sem_label` links between semantic labels and lexical units: only lexical units that were headwords of DiCo records possess a semantic labeling. Statistics above show that the LS contains 1,690 lexical units. So where do the 915 (1,690 – 775) extra units come from? They all have been extrapolated from the so-called phraseology (`ph`) field of DiCo records, where lexicographers list idioms that are formally built from the record headword. For instance, the DiCo record for BARBE 'beard' contained (among others) a pointer to the idiom BARBE À PAPA 'cotton candy.' This idiom did not possess its own record in the original DiCo and has been "reified"

[4] http://www.olst.umontreal.ca/dicouebe.

[5] The code for compiling the DiCo into an LS, generating GraphML exports and generating an HTML version of the DiCo has been written in SWI-Prolog.

[6] The hierarchy of semantic labels is developed with the Protégé ontology editor. We use XML exports from Protégé to inject this hierarchy inside the LS. This is another illustration of the cannibalistic (and not too choosy) nature of LSs.

while generating the LS, among 914 other idioms.

The "wordlist" of our LS is therefore much more developed than the wordlist of the DiCo it is derived from. This is particularly true if we include in it the 6,464 wordform entities. As explained earlier, it is possible to regenerate from the LS lexical descriptions for any lexical entity that is either a lexical unit or a wordform targeted by a lexical function application, filling wordform descriptions with inverse lexical function links. To test this, we have regenerated an entire DiCo in HTML format from the LS, with a total of 8,154 (1,690 + 6,464) lexical entries, stored as individual HTML pages. Pages for original DiCo headwords contain the hypertext specification of the original lexical function links, together with all inverse lexical links that have been found in the LS; pages for wordforms contain only inverse links. For instance, the page for METTRE 'to put' (which is not a headword in the original DiCo) contains 71 inverse links, such as:[7]

```
CausOper1( À L'ARRIÈRE-PLAN# ) ->
Labor12( ACCUSATION#I.2 ) ->
Caus1[1]Labreal1( ANCRE# ) ->
Labor21( ANGOISSE# ) ->
Labreal12( ARMOIRE# ) ->
```

Of course, most of the entries that were not in the original DiCo are fairly poor and will require significant editing to be turned into *bona fide* DiCo descriptions. They are, however, a useful point of departure for lexicographers; additionally, the richer the DiCo will become, the more productive the LS will be in terms of automatic generation of draft descriptions.

5 Lexical systems and multilinguality

The approach to multilingual implementation of lexical resources that LSs allow is compatible with strategies used in known multilingual databases, such as Papillon (Sérasset and Mangeot-Lerebours, 2001): it sees multilingual resources as connections of basically monolingual models. In this final section, we first argue for a monolingual perspective on the problem of multilinguality. We then make proposals for implementing interlingual connections by means of LSs.

5.1 Theoretical and methodological primacy of monolingual structures

We see two logical reasons why the issue of designing multilingual lexical databases should be tackled from a monolingual perspective.

First, all natural languages can perfectly well be conceived of in complete isolation. In fact, monolingual speakers are no less "true" speakers of a language than multilingual speakers.

Second, acquisition of multiple languages commonly takes place in situations where **second** languages are acquired as additions to an already mastered first language. Multiplicity in linguistic competence is naturally implemented by graft of a language on top of a preexisting linguistic knowledge. How multiple lexica are acquired and stored is a much debated issue (Schreuder and Weltens, 1993), which is outside the scope of our research. However, it is now commonly accepted that even children who are bilingual "from birth" develop two linguistic systems, each of which being quite similar in essence to linguistic systems of monolingual speakers (de Houwer, 1990). The main issue is thus one of systems' connectivity.

From a theoretical and practical point of view, it is thus perfectly legitimate to see the problem of structuring multilingual resources as one of, first, finding the most adequate and interoperable structuring for monolingual resources. This being said, we do not believe that the issue of structuring monolingual databases has already been dealt with once and for all in a satisfactory manner. We hope the concept of LS we introduce here will stimulate reflection on that topic.

5.2 Multilingual connections between LSs

A multilingual lexical resource based on the LS architecture should be made up of several **fully autonomous LSs**, i.e., LSs that are not specially tailored for multilingual connections. They should function as independent modules that can be connected while preserving their integrity.

Connections between LSs should be implemented as specialized interlingual links between equivalent lexical entities. There is one exception however: standard lexical functions (`A1`, `Magn`, `AntiMagn`, `Oper1`, etc.). Because they are universal lexical entities, they should be stored in a specialized interlingual module; as universals, they play a central role in interlingual connectivity (Fontenelle, 1997). However, these are only "pure" lexical functions. Lexical function appli-

[7] We underline hypertext links. Lexical function applications listed here correspond French collocations that mean, respectively, *to put in the background*, *to indict someone* (literally in French 'to put someone in accusation'), *to anchor a vessel* (literally in French 'to put a vessel at the anchor'), *to put someone in anguish*, *to keep something in a cupboard*.

cations, such as **Oper12**(RANCUNE) above, are by no means universals and have to be connected to their counterpart in other languages. Let us examine briefly this aspect of the question.

One has to distinguish at least two main cases of interlingual lexical connections in LSs: direct lexical connections and connections through lexical function applications.

Direct connections, such as Fr. RANCUNE *vs.* Eng. RESENTMENT should be implemented—manually or using existing bilingual resources—as simple interlingual (i.e. intermodule) links between two lexical entities. Things are not always that simple though, due to the existence of partial or multiple interlingual connections. For instance, what interlingual link should originate from Eng. SIBLING if we want to point to a French counterpart? As there is no lexicalized French equivalent, we may be tempted to include in the French LS entities such as *frère ou sœur* ('brother or sister'). We have two strong objections to this. First, this complex entity will not be a proper translation in most contexts: one cannot translate *He killed all his siblings* by *Il a tué tous ses frères ou sœurs*—the conjunction *et* 'and' is required in this specific context, as well as in many others. Second, and this is more problematic, this approach would force us to enter in the French LS entities for translation purposes, which would transgress the original monolingual integrity of the system.[8] We must admit that we do not have a ready-to-use solution to this problem, specially if we insist on ruling out the introduction of *ad hoc* periphrastic translations as lexical entities in target LSs. It may very well be the case that a cluster of interrelated LSs cannot be completely connected for translation purposes without the addition of "buffer" LSs that ensure full interlingual connectivity. For instance, the buffer French LS for English to French LS connection could contain phrasal lexical entities such as *frères et sœurs* ('siblings'), *être de mêmes parents* and *être frère*(*s*) et *sœur*(*s*) ('to be siblings'). This strategy can actually be very productive and can lead us to realize that what appeared first as an *ad hoc* solution may be fully justified from a linguistic perspective. Dealing with the *sibling* case, for instance, forced us to realized that while *frère*(*s*) *et sœur*(*s)* sounds very normal in French, *sœur*(*s*) *et frère*(*s*) will seem odd or, at least, intentionally built that way. This is a very strong argument for considering that a lexical entity (we do not say *lexical unit*!) *frère*(*s*) *et sœur*(*s*) **does** exist in French, independently from the translation problem that *sibling* poses to us. This phrasal entity should probably be present in any complete French LS.

The case of connections through lexical function applications is even trickier. A simplistic approach would be to consider that it is sufficient to connect interlinguistically lexical function applications to get all resulting lexical connections for value elements. For standard lexical functions, this can be done automatically using the following strategy for two languages A and B.

> If the lexical entity L_A is connected to L_B by means of a "translation" link,
> all lexical entities linked to the lexical function application **f**(L_A) by the "value" link should be connected by a "value translation" link, with a trust value of "0.5," to all lexical entities linked to **f**(L_B) by a "value" link.

The distinction between "translation" and "value translation" links allow for contextual interlingual connections: a lexical entity L'_B could happen to be a proper translation of L'_A only if it occurs as collocate in a specific collocation. But this is not enough. It is also necessary to filter "value translation" connections that are systematically generated using the above strategy. For instance, each of the specific values given in (1) section 3 should be associated with its **closest** equivalent among values of **Oper12**(RESENTMENT): HAVE, FEEL, HARBOR, NOURISH, etc. At the present time, we do not see how this can be achieved automatically, unless we can make use of already available multilingual databases of collocations. For English and French, for instance, we plan to experiment in the near future with T. Fontenelle's database of English-French collocation pairs (Fontenelle, 1997). These collocations have been extracted from the *Collins-Robert* dictionary and manually indexed by means of lexical functions. We are convinced it is possible to use this database firstly to build a first version of a new English LS and, secondly, to implement the type fine-grained multilingual connections between lexical function values illustrated with our RANCUNE *vs.* RESENTMENT example.

We are well aware that we have probably surfaced as many problems as we have offered solutions in this section. However, the above considerations show at least two things:

[8] It is worth noticing that good English-French dictionaries, such as the *Collins-Robert*, offer several different translations in this particular case. Additionally, their translations do not apply to *sibling* as such, but rather to *siblings* or to expressions such as *someone's siblings*, *to be siblings*, etc.

- LSs have the merit to make explicit the scale of the problem of interlingual lexical correspondence, if one want to tackle this problem in a fine-grained manner;[9]
- the implementation of multilingual connections over LSs should be approached using semi-automatic strategies.

6 Conclusions

We have achieved the production of a significant LS, which can be considered of broad coverage in terms of the sheer number of entities and links it contains and the richness of linguistic knowledge it encodes. We plan to finish the absorption of all information contained in the dictionary-like DiCo (including information that can be inferred). We also want to integrate complementary French databases into the LS (for instance the Morphalou database,[10] for morphological information) and start to implement multilingual connections using T. Fontenelle's collocation database. Another development will be the construction of an editor to access and modify the content of our LS. This tool could also be used to develop DiCo-style LSs for other languages than French.

Acknowledgments

This research is supported by the *Fonds québécois de la recherche sur la Société et la culture* (FQRSC). We are very grateful to Sylvain Kahane, Marie-Claude L'Homme, Igor Mel'čuk, Ophélie Tremblay and four MLRI 2006 anonymous reviewers for their comments on a preliminary version of this paper. A very special thank to Sylvain Kahane and Jacques Steinlin for their invaluable work on the DiCo SQL, that made our own research possible.

References

American Heritage. 2000. *The American Heritage Dictionary of the English Language.* Fourth Edition, CD-ROM, Houghton Mifflin, Boston, MA.

Jean Aitchison. 2003. *Words in the Mind: An Introduction to the Mental Lexicon*, 3rd edition, Blackwell, Oxford, UK.

Collin F. Baker, Charles J. Fillmore and Beau Cronin. 2003. The Structure of the Framenet Database. *Int. Journal of Lexicography*, 16(3): 281-296.

James W. Breen. 2004. JMdict: a Japanese-Multilingual Dictionary. *Proceedings of COLING Multilingual Linguistic Resources Workshop*, Geneva, Switzerland.

Annick de Houwer. 1990. *The Acquisition of Two Languages from Birth. A Case Study*, Cambridge University Press, Cambridge, UK.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*, MIT Press, Cambridge, MA.

Thierry Fontenelle. 1997. *Turning a bilingual dictionary into a lexical-semantic database*, Niemeyer, Tübingen, Germany.

Sylvain Kahane and Alain Polguère. 2001. Formal foundation of lexical functions. *Proceedings of ACL/EACL 2001 Workshop on Collocation*, Toulouse, France, 8-15.

François Lareau. 2002. A Practical Guide for Writing DiCo Entries. *Third Papillon 2002 Seminar*, Tokyo, Japan [http://www.papillon-dictionary.org/ConsultInformations.po?docid=1620757&docLang=eng].

Igor Mel'čuk. 1996. Lexical Functions: A Tool for the Description of Lexical Relations in the Lexicon. In Leo Wanner (ed.): *Lexical Functions in Lexicography and Natural Language Processing*, Amsterdam/ Philadelphia: Benjamins, 37-102.

Igor Mel'čuk, André Clas and Alain Polguère. 1995. *Introduction à la lexicologie explicative et combinatoire*, Duculot, Louvain-la-Neuve, Belgium.

Igor Mel'čuk and Leo Wanner. 2001. Towards a Lexicographic Approach to Lexical Transfer in Machine Translation (Illustrated by the German-Russian Language Pair). *Machine Translation*, 16: 21-87.

Alain Polguère. 2000. Towards a theoretically-motivated general public dictionary of semantic derivations and collocations for French. *Proceedings of EURALEX'2000*, Stuttgart, Germany, 517-527.

Robert Schreuder and Bert Weltens (eds.). 1993. *The Bilingual lexicon*, Amsterdam, Benjamins.

Gilles Sérasset and Mathieu Mangeot-Lerebours. 2001. Papillon lexical database project: Monolingual dictionaries and interlingual links. *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium*, Tokyo, Japan, 119–125.

Jacques Steinlin, Sylvain Kahane and Alain Polguère. 2005. Compiling a "classical" explanatory combinatorial lexicographic description into a relational database. *Proceedings of the Second International Conference on the Meaning Text Theory*, Moscow, Russia, 477-485.

[9] For a systematic analysis of interlingual lexical correspondences, see Mel'čuk and Wanner (2001).
[10] http://actarus.atilf.fr/morphalou/