Using an open-source unification-based system for CL/NLP teaching

Ann Copestake
Computer Laboratory
University of Cambridge
Cambridge, UK
aac@cl.cam.ac.uk

John Carroll
Cognitive and Computing Sciences
University of Sussex
Falmer, Brighton, UK
johnca@cogs.susx.ac.uk

Dan Flickinger
CSLI, Stanford University *and*
YY Software
Ventura Hall,
Stanford, USA
danf@csli.stanford.edu

Robert Malouf
Alfa Informatica,
University of Groningen,
Postbus 716, 9700 AS Groningen,
The Netherlands
malouf@let.rug.nl

Stephan Oepen
YY Software *and*
CSLI, Stanford University
110 Pioneer Way
Mountain View, USA
oe@yy.com

Abstract

We demonstrate the open-source LKB system which has been used to teach the fundamentals of constraint-based grammar development to several groups of students.

1 Overview of the LKB system

The LKB system is a grammar development environment that is distributed as part of the open source LinGO tools (`http://www-csli.stanford.edu/~aac/lkb.html` and `http://lingo.stanford.edu`, see also Copestake and Flickinger, 2000). It is an open-source grammar development environment implemented in Common Lisp, distributed not only as source but also as a standalone application that can be run on Linux, Solaris and Windows (see the website for specific requirements). It will also run under Macintosh Common Lisp, but for this a license is required. The LKB includes a parser, generator, support for large-scale inheritance hierarchies (including the use of defaults), various tools for manipulating semantic representations, a rich set of graphical tools for analyzing and debugging grammars, and extensive on-line documentation. Grammars of all sizes have been written using the LKB, for several languages, mostly within the linguistic frameworks of Categorial Grammar and Head-Driven Phrase Structure Grammar. The LKB system was initially developed in 1991, but has gone through multiple versions since then. It is in active use by a considerable number of researchers worldwide. An introductory book on implementing grammars in typed feature structure formalisms using the LKB is near completion (Copestake, in preparation).

2 Demo outline

Although the LKB has been successfully used for large-scale grammar development, this demonstration will concentrate on its use with relatively small scale teaching grammars, of a type which can be developed by students in practical exercises. We will show an English grammar fragment which is linked to a textbook on formal syntax (Sag and Wasow, 1999) to illustrate how the system may be used in conjunction with more traditional materials in a relatively linguistically oriented course. We will demonstrate the tools for analyzing parses and for debugging and also discuss the way that parse selection mechanisms can be incorporated in the system. If time permits, we will show how semantic analyses produced with a somewhat more complex grammar can be linked up to a theorem prover and also exploited in semantic transfer for Machine Translation. Exercises where the grammar is part of a larger system are generally appropriate for advanced courses or for NLP application courses.

The screen dump in the figure is from a session working with a grammar fragment for Esperanto. This shares its basic types and rules with the English textbook grammar fragment mentioned above. The windows shown are:

1. The LKB Top interaction window: main

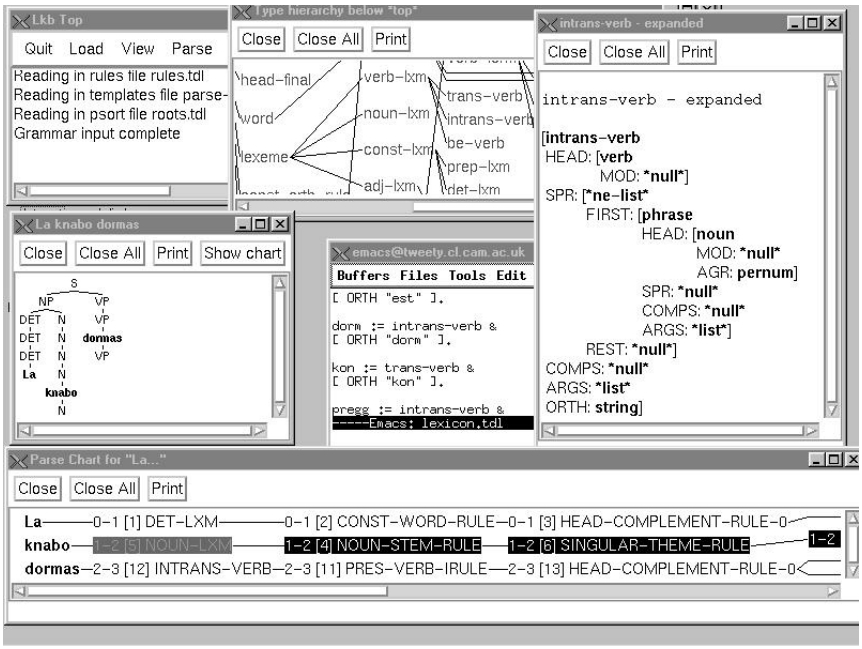

Figure 1: Screen dump of the LKB system

menus plus feedback and error messages

2. Type hierarchy window (fragment): the more general types are on the left. Nodes in the hierarchy have menus that provide more information about the types, such as their associated constraints.

3. Type constraint for the type **intrans-verb**: again nodes are clickable for further information.

4. Parse tree for *La knabo dormas* (*the boy sleeps*): a larger display for parse trees is also available, but this scale is useful for summary information. Menus associated with trees allow for display of associated semantic information if any is included in the grammar and for generation. Here the display shows inflectional rules as well as normal syntactic rules: hence the VP node under *dormas*, which corresponds to the stem.

5. In the middle is an emacs window displaying the source file for the lexicon associated with this grammar.[1] It shows the entry for the lexeme *dorm*, which, like most lexical entries in this grammar, just specifies a spelling and a type (here **intrans-verb**).

6. Part of the parse chart corresponding to the tree is shown in the bottom window: nodes which have *knabo* as a descendant are highlighted. Again, these nodes are active: one very useful facility associated with them is a unification checker which allows the grammar writer to establish why a rule did not apply to a phrase or phrases.

3 Use of the LKB in teaching

Teaching uses of the LKB have included undergraduate and graduate courses on formal syntax and on computational linguistics at several sites, grammar engineering courses at two ESSLLI summer schools, and numerous student projects at undergraduate, masters and doctoral levels. An advantage of the LKB is that students learn to use a system which is sufficiently heavy duty for more advanced work, up to the scale at least of research

[1](We generally use emacs as an editor when teaching, although this causes some overhead, especially for students who are only used to word processing programs.

prototypes. This provides them with a good platform on which to build for further research. Feedback from the courses we have taught has mostly been very positive, but we have found a ratio of six students to one instructor (or teaching assistant) to be the maximum that is workable. One major reason is that debugging students' grammars and teaching debugging techniques is time-consuming.

When teaching an introductory course with the LKB, we start the students off with a very simple grammar, which they are asked to expand in specific ways. We introduce various additional techniques and formal devices (such as inflectional and lexical rules, defaults, difference lists and gaps) gradually during a course. Material from our ESSLLI courses, including starting grammars, exercises and solutions is distributed via the website. Several other small grammars developed by students are also distributed as part of the LKB system and we would welcome further contributions. We are hoping to facilitate this by making it easier for people outside the LinGO group to add and modify grammars.

Several graduate students have used versions of the LKB system as part of their thesis work, for diverse projects including machine translation and grammar learning. It has been used in the development of several large grammars, especially the LinGO English Resource Grammar (ERG), which is itself open-source. Research applications for the ERG include spoken language machine translation in Verbmobil, generation for a speech prosthesis, and automated email response, under development for commercial use. The LKB/ERG combination can be used by researchers who require a grammar which provides a detailed semantic analysis and reasonably broad coverage, for instance for experiments on dialogue. The LKB has also been used as a grammar preprocessor to facilitate experiments on efficiency using the ERG with other systems (Flickinger et al, 2000).

4 Comparison with other work

There is a long history of the use of feature structure based systems in teaching, dating back at least to PATR (Shieber, 1986: see `http://www.ling.gu.se/~li/`). The Alvey Natural Language Tools (Briscoe et al, 1987) have been used for teaching at several universities: Briscoe and Grover developed an extensive set of teaching examples and exercises, which is however unpublished. Versions of the SRI Core Language Engine (Alshawi, 1992) and of the XTAG grammar (XTAG group, 1995) and parser have also been used for teaching. Besides the LKB, typed feature structure environments have been used at many universities, though unlike the systems cited above, most have only been used with small grammars and may not scale up. Hands on courses using various systems have been run at many recent summer schools including ESSLLI 99 (using the Xerox XLE, see Butt et al, 1999) and ESSLLI 97 and the 1999 LSA summer school (both using ConTroll, see Hinrichs and Meurers, 1999). Very little seems to have been formally published describing experiences in teaching with grammar development environments, though Bouma (1999) describes material for teaching a computational linguistics course that includes exercises using the Hdrug unification-based enviroment to extend a grammar.

Despite this rich variety of tools, we believe that the LKB system has a combination of features which make it distinctive and give it a useful niche in teaching. The most important points are that its availability as open source, combined with scale and efficiency, allow advanced projects to be supported as well as introductory courses. As far as we are aware, it is the only system freely available with a broad coverage grammar that supports semantic interpretation and generation. Especially for more linguistically oriented courses, the link to the Sag and Wasow textbook is also important. Similar grammars could be developed for other systems, but would be less directly comparable to the textbook since this assumes a default formalism which so far is only implemented in the LKB.

On the other hand, the LKB is not a suitable basis for a course that involves the students learning to implement a unifier, parser and so on. The system is quite complex (about 120 files and 40,000 lines of Lisp code) and though the vast majority of this is concerned with non-core functionality, such as the graphical interfaces, it is still some-

what daunting. This seems an inevitable trade-off of having a system powerful enough for real applications (see Bouma (1999) for related discussion). It is questionable whether the LKB is entirely satisfactory as a student's first computational grammar system, although we have used it with students who have no prior experience of this sort: ideally we would suggest starting off with brief exercises with a pure context-free grammar to explain the concepts of well-formedness, recursion and so on. We also wouldn't necessarily advocate using the LKB as a core component of a first course on formal syntax for linguistic students, since the specifics of dealing with an implementation may interfere with understanding of basic concepts, though it is suitable as a supplement to an initial course or as the basis for a slightly more advanced course.

We think there is considerable potential for building materials for courses that allow students to work with realistic but transparent applications using the LKB and a large grammar as a component. Developing such materials is clearly necessary in order to give students useful practical experience. It is however very time-consuming, and most probably will have to be undertaken as part of a cooperative, open-source development involving people from several different institutions.

Acknowledgements

This research was partially supported by the National Science Foundation, grant number IRI-9612682. The current versions of the English grammars associated with the Sag and Wasow textbook were largely developed by Christopher Callison-Burch while he was an undergraduate at Stanford.

References

Alshawi, Hiyan (ed). [1992] *The Core Language Engine*, MIT Press, Cambridge, MA.

Bouma, Gosse. [1999] 'A modern computational linguistics course using Dutch.' In Frank van Eynde and Ineke Schuurman, editors, CLIN 1998, Papers from the Ninth CLIN Meeting, Amsterdam. Rodopi Press.

Briscoe, Ted, Claire Grover, Bran Boguraev and John Carroll. [1987] 'A formalism and environment for the development of a large grammar of English', *Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI-87)*, Milan, Italy, 703–708.

Butt, Miriam, Anette Frank and Jonas Kuhn. [1999] 'Development of large scale LFG grammars – Linguistics, Engineering and Resources', `http://www.xrce.xerox.com/people/frank/esslli99-hp/index.html`

Copestake, Ann. [in preparation] *Implementing typed feature structure grammars*, CSLI Publications, Stanford.

Copestake, Ann and Dan Flickinger. [2000] 'An open-source grammar development environment and broad-coverage English grammar using HPSG', Second conference on Language Resources and Evaluation (LREC-2000), Athens, Greece.

Flickinger, Daniel, Stephan Oepen, Hans Uszkoreit and Jun'ichi Tsujii. [2000] *Journal of Natural Language Engineering. Special Issue on Efficient Processing with HPSG: Methods, Systems, Evaluation*, 6(1).

Hinrichs, Erhard and Detmar Meurers [1999] 'Grammar Development in Constraint-Based Formalisms', `http://www.ling.ohio-state.edu/~dm/lehre/lsa99/material.html`, see also `http://www.sfs.nphil.uni-tuebingen.de/controll/`

Sag, Ivan, and Tom Wasow [1999] *Syntactic Theory: An Introduction*, CSLI Publications.

Shieber, Stuart [1986] *An Introduction to Unification-based Approaches to Grammar*, CSLI Publications.

The XTAG Research Group [1995]. 'A Lexicalized Tree Adjoining Grammar for English' IRCS Report 95-03, University of Pennsylvania"