Generating an LTAG out of a principle-based hierarchical representation

Marie-Hélène Candito
TALANA and UFRL, Université Paris 7
2, place Jussieu, Tour centrale 8ème étage pièce 801
75251 Paris cedex 05 FRANCE
marie-helene.candito@linguist.jussieu.fr

Abstract

Lexicalized Tree Adjoining Grammars have proved useful for NLP. However, numerous redundancy problems face LTAGs developers, as highlighted by Vijay-Shanker and Schabes (92).

We present and a tool that automatically generates the tree families of an LTAG. It starts from a compact hierarchical organization of syntactic descriptions that is linguistically motivated and carries out all the relevant combinations of linguistic phenomena.

1 Lexicalized TAGs

Lexicalized Tree Adjoining Grammar (LTAG) is a formalism integrating lexicon and grammar (Joshi, 87; Schabes et al., 88), which has proved useful for NLP. Linguists have developed over the years sizeable LTAG grammars, especially for English (XTAG group, 95) and French (Abeillé, 91).

In this formalism, the lexical items are associated with elementary trees representing their maximal projection. Features structures are associated with the trees, that are combined with substitution and adjunction. Adjunction allows the extended domain of locality of the formalism : all trees anchored by a predicate contain nodes for all its arguments.

Such a lexicalized formalism needs a practical organization. LTAGs consist of a morphological lexicon, a syntactic lexicon of lemmas and a set of tree schemata, i.e. trees in which the lexical anchor is missing. In the syntactic lexicon, lemmas select the tree schemata they can anchor[1].

The set of tree schemata forms the syntactic part of the grammar. The tree schemata selected by predicative items are grouped into families, and collectively selected. A tree family contains the different possible trees for a given canonical subcategorization. Along with the "canonical" trees, a family contains the ones that would be transformationally related in a movement-base approach. These are first the trees where a “redistribution” of the syntactic function of the arguments has occurred, for instance the passive trees, or middle (for French) or dative shift (for English), leading to an "actual subcategorization" different from the canonical one. Secondly, a family may contain the trees with extracted argument (or cliticized in French).

In the syntactic lexicon, a particular lemma may select a family only partially. For instance a lemma might select the transitive family, ruling out the passive trees. On the other hand, the features appearing in the tree schemata are common to every lemma selecting these trees. The idiosyncratic features (attached to the anchor or upper in the tree) are introduced in the syntactic lexicon.

2 Development and maintenance problems with LTAGs

This extreme lexicalization entails that a sizeable LTAG comprises hundreds of elementary trees (over 600 for the cited large grammars). And as highlighted by Vijay-Shanker and Schabes (92), information on syntactic structures and associated features equations is repeated in dozens of tree schemata (hundreds for subject-verb agreement for instance).

Redundancy makes the tasks of LTAG writing, extending or updating very difficult, especially because all combinations of phenomena must be handled. And, in addition to the practical problem of grammar storage, redundancy makes it hard to get a clear vision of the theoretical and practical choices on which the grammar is based.

3 Existing solutions

A few solutions have been proposed for the problems described above. They use two main devices for lexicon representation : inheritance networks and lexical rules. But for LTAG representation, inheritance networks have to include phrase-structure information also, and lexical rules become “lexico-syntactic rules”. Vijay-Shanker and Schabes, (92) have first proposed a scheme for LTAG representation. Implemented work is also described in (Becker, 93; 95) and (Evans et al., 95).

The three cited solutions give an efficient representation (without redundancy) of an LTAG, but have in our opinion two major deficiencies. First these solutions use inheritance networks and lexical rules in a purely technical way. They give no principle about the form of the hierarchy or the lexical rules, whereas we

[1] At grammar use, the words of the sentence to be parsed are associated with the relevant tree schemata to form complete lexicalized trees.

believe that addressing the practical problem of redundancy should give the opportunity of formalizing the well-formedness of elementary trees and of tree families. And second, the *generative* aspect of these solutions is not developed. Certainly the lexical rules are proposed as a tool for generation of new schemata or new classes in a inheritance network. But the *automatic* triggering, ordering and bounding of the lexical rules is not discussed[2].

4 Proposed solution : a principle-based representation and a generation system

We propose a system for the writing and/or the updating of an LTAG. It comprises a principled and hierarchical representation of lexico-syntactic structures. Using this hierarchy and principles of well-formedness, the tool carries out automatically the relevant crossings of linguistic phenomena to generate the tree families.

This solution not only addresses the problem of redundancy but also gives a more principle-based representation of an LTAG. The implementation of the principles gives a real generative power to the tool.

Due to a lack of space we cannot develop all the aspects of this work[3]. After a brief description of the organization of the syntactic hierarchy, we will focus on the use of partial descriptions of trees.

4.1 Organization of the hierarchy

The proposed organization of the hierarchy follows from the linguistic principles of well-formedness of elementary TAG trees, mainly the predicate-arguments co-occurrence principle (Kroch and Joshi, 85; Abeillé, 91) : the trees for a predicative item contain positions for all its arguments.

But for a given predicate, we expect the canonical arguments to remain constant through redistribution of functions. The canonical subject (argument 0) in a passive construction, even when unexpressed, is still an argument of the predicate. So the principle should be a principle of predicate-*functions* co-occurrence : the trees for a predicative item contain positions for all the functions of its actual subcategorization.

This reformulated principle presupposes the definition of an actual subcategorization, given the canonical subcategorization of a predicate. This presupposition and the predicate-functions co-occurrence principle are fulfilled by organizing the hierarchy along the three following dimensions :

dimension 1 : canonical subcategorization frame
This dimension defines the types of canonical subcategorization. Its classes contain information on the arguments of a predicate, their index, their possible categories and their canonical syntactic function.

dimension 2 : redistribution of syntactic functions
This dimension defines the types of redistribution of functions (including the case of no redistribution at all). The association of a canonical subcategorization frame and a compatible redistribution gives an actual subcategorization, namely a list of argument-function pairs, that have to be locally realized.

dimension 3 : syntactic realizations of functions
It expresses the way the different syntactic functions are positioned at the phrase-structure level (in canonical, cliticized, extracted position...).

These three dimensions constitute the core hierarchy. Out of this syntactic database and following principles of well-formedness the generator creates elementary trees. This is a two-steps process : it first creates some *terminal classes* with inherited properties only - they are totally defined by their list of super-classes. Then it translates these terminal classes into the relevant elementary tree schemata, in the XTAG[4] format, so that they can be used for parsing.

Tree schemata generation respects the predicate-functions co-occurrence principle. Their corresponding terminal classes are created first by associating a canonical subcat (dimension 1) with a compatible redistribution, including the case of no redistribution (dimension 2). Then for each function defined in the actual subcat, exactly one realization of function is picked up in dimension 3.

The generation is made family by family. This is simply achieved by fixing the canonical subcat frame (dimension 1), At the development stage, generation can also be done following other criterions. For instance, all passive trees or all trees with extracted complements can be generated.

4.2 Formal choices : monotonic inheritance network and partial descriptions of trees

The generation process described above is quite powerful in the context of LTAGs, because it carries out automatically all the relevant crossings of linguistic phenomena. These crossings are precisely the major source of redundancy in LTAGs. Because of this generative device, we do not need to introduce lexico-syntactic rules, and thus we do not have to face the problems of ordering and bounding their application.

Further, as was mentioned in section 1, lexical idiosyncrasies are handled in the syntactic lexicon, and not in the set of tree schemata. So to represent hierarchically this set, we do not think that nonmonotonicity is linguistically justified. We have thus chosen monotonicity, which gives more transparency and improves declarativity. We follow here

[2] Becker (93) gives a linguistic principle for the bounding of his meta-rules, but has no solution for the application of this principle.

[3] A fuller description of the work can be found in (Candito, to appear)

[4] XTAG (Paroubek et al., 92) is a tool for writing and using LTAGs, including among other things a tree editor and a syntactic parser.

Vijay-Shanker and Schabes (92) and use partial descriptions of trees (Rogers and Vijay-Shanker, 94)[5].

A partial description is a set of constraints that characterizes a set of trees. Adding information to the description reduces monotonically the set of satisfying trees. The partial descriptions of Rogers and Vijay-Shanker (94) use three relations : left-of, parent and dominance (represented with a dashed line). A dominance link can be further specified as a path of length superior or equal to zero. These links are obviously useful to underspecify a relation between two nodes at a general level, that will be specified at an either lower or lateral level. Figure 1 shows a partial description representing a sentence with a nominal subject in canonical position, giving no other information about possible other complements. The underspecified link between the S and V nodes allows for either presence or absence of a cliticized complement on the verb. In the case of a clitic, the path between the S and V nodes can be specified with the description of figure 2. Then, if we have the information that the nodes labelled respectively S and V of figures 1 and 2 are the same, the conjunction of the two descriptions is equivalent to the description of figure 3.

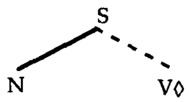

Figure 1

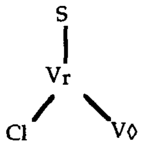

Figure 2

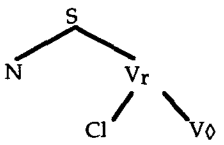

Figure 3

This example shows the declarativity obtained with partial descriptions that use large dominance links. The inheritance of descriptions of figure 1 and 2 is order independent. Without large dominance links, an order of inheritance of the classes describing a subject in canonical position and a cliticized complement should be predefined.

In the hierarchy of syntactic descriptions we propose, the partial description associated with a class is the unification of the own description of the class with all inherited partial descriptions. Identity of nodes is stated in our system by "naming" both nodes in the same way, since in descriptions of trees, nodes are referred to by constants. Two nodes, in two conjunct descriptions, referred to by the same constant are the same node. Equality of nodes can also be inferred, mainly using the fact that a tree node has only one direct parent node.

We have added atomic features associated with each constant, such as category, index, canonical syntactic function and actual syntactic function. In the conjunction of two descriptions, the identification of two nodes known to be the same requires the unification of such features. In case of failure, the whole conjunction leads to an unsatisfiable description.

A terminal class is translated into its corresponding elementary tree(s) by taking the minimal satisfying tree(s) of the partial description of the class[6].

4.3 Application to the French LTAG

The tool was used to generate tree families of the French grammar, using a hand-written hierarchy of syntactic descriptions. This task is facilitated by the guidelines given on the form of the hierarchy. Out of about 90 hand-written classes, the tool generates 730 trees for the 17 families for verbs without sentential complements[7], 400 of which were present in the pre-existing grammar. We have added phenomena such as some causative constructions or free order of complements.

The proposed type of hierarchy is meant to be universal, and we are currently working on its application to Italian.

5 References

A. Abeillé. 1991. Une grammaire lexicalisée d'Arbres Adjoints pour le français, PhD thesis, Univ. Paris 7.

T. Becker. 1993. HyTAG : a new type of Tree Adjoining Grammars for Hybrid Syntactic Representation of Free Order Languages, PhD thesis, Univ. of Saarbrücken.

T. Becker. 1994. Patterns in Metarules. Proc. of TAG+3.

M-H. Candito. To appear. A principle-based hierarchical representation of LTAGs. Proc. of COLING'96, Copenhagen.

R. Evans, G. Gazdar and D. Weir. 1995. Encoding Lexicalized Tree Adjoining Grammar with a Nonmonotonic Inheritance Hierarchy. Proc. of ACL'95, Boston.

A. Joshi. 1987. Introduction to Tree Adjoining Grammar, in A. Manaster Ramer (ed), *The Mathematics of Language*, J. Benjamins, pp. 87-114.

A. Kroch and A. Joshi. 1985. The linguistic relevance of Tree Adjoining Grammars. Technical report, Univ. of Pennsylvania.

P. Paroubek, Y. Schabes and A. Joshi. 1992. XTAG - A graphical Workbench for developing Tree Adjoining Grammars. Proc. of 3-ANLP, Trento.

J. Rogers and K. Vijay-Shanker. 1992. Reasoning with descriptions of trees. Proc. ACL'92, pp. 72-80.

J. Rogers and K. Vijay-Shanker. 1994. Obtaining trees from their descriptions : an application to Tree-Adjoining Grammars. Computational Intelligence, vol. 10, N° 4.

Y. Schabes, A. Abeillé and A. Joshi. 1988. Parsing strategies with lexicalized grammars : Tree Adjoining Grammars. Proc. of COLING'88, Budapest, vol. 2.

K. Vijay-Shanker and Y. Schabes. 1992. Structure Sharing in Lexicalized Tree Adjoining Grammar. Proc. of COLING'92, Nantes, pp. 205-211.

XTAG research group. 1995. A lexicalized TAG for English, Technical Report IRCS 95-03, Univ. of Pennsylvania.

[5] Vijay-Shanker & Schabes (92) have used the partial descriptions introduced in (Rogers & Vijay-Shanker, 92), but we have used the more recent version of (Rogers & Vijay-Shanker, 94). The difference lies principally in the definition of quasi-trees, first seen as partial models of trees and later as distinguished sets of constraints.

[6] Intuitively the remaining underspecified links are taken to be path of minimal length. See Rogers and Vijay-Shanker (94).

[7] By the time of conference, we will be able to give figures for the families with sentential complements also.