Turning Elementary Trees into Feature Structures

Alexandra Kinyon
University of Pennsylvania
Institute for Research in Cognitive Science
Suite 400A, 3401 Walnut Street
Philadelphia, PA 19104-6228
kinyon@linc.cis.upenn.edu
http://www.cis.upenn.edu/∼kinyon

1. Introduction

The richness of the TAG formalism allows one to encode a wide variety of linguistic information in the topology of an elementary tree (ex: argument vs adjunct distinction, information about "belonging" to a given subcategorization class, constraints on word order, realization of one or more syntactic phenomena such as passive, wh extraction etc...). Given a number of "linguistic" information and constraints applying to lexical items, a grammar developer will encode this information into one or several elementary trees. However, proceeding in the opposite direction is far less obvious : given an elementary tree, which crucial pieces of linguistic information are provided by this tree, and how can these pieces be clearly isolated and represented ?

In this paper we try to answer that question. More precisely, we investigate the different possibilities for extracting from each elementary tree of a TAG a feature structure which captures and clearly separates each piece of essential linguistic information conveyed by that elementary tree. [1]

We examine **existing implementations** of wide-coverage TAG grammars. Current existing wide-coverage TAG grammars are either semi-automatically generated from an abstract level of syntactic representation such as MetaRules or a MetaGrammar, or are "plain" grammars that do not resort to such an intermediate step of abstract syntactic generalization [2]. Therefore, after explaining in the first part of this paper why featurizing a TAG is useful, we then discuss the featurization of these three types of TAGs: We examine the "featurization" of a TAG semi-automatically generated with a MetaGrammar, the featurization of a TAG generated with MetaRules, and finally, several options for featurizing a plain TAG (hand-crafted, or automatically acquired from a treebank). We argue that automatic featurization is desirable, and show that such an automation is achievable thanks to existing implementations, either via a MetaGrammar approach or via a MetaRules approach, even in order to featurize plain grammars.

2. Why featurize a TAG ?

By featurizing a elementary tree, we mean capturing its main linguistic characteristics and representing them in a feature structure. Of course, several kinds of information may be envisioned, depending on the application. Still, a general need for featurization appears for tasks as diverse as improving Supertagging accuracy, Text Generation, Corpus annotation and search, and extraction of framework independent linguistic information.

2.1. Improving Supertagging accuracy

Supertagging consists in assigning one or several elementary tree(s) to each lexical item in a text (Srinivas, 1997). Good SuperTagging accuracy is obtained with medium-size grammars[3]. However, as the size of a TAG increases from a few hundred trees for hand-crafted grammars to a few thousand trees for automatically generated grammars, the accuracy of Supertagging deteriorates (just as for traditional POS tagging, the POS accuracy

1. Since we are concerned with *linguistic* information, we leave aside approaches which focus on capturing common *mathematical* or *topological* properties of elementary trees such as those discussed in (Kallmeyer, 1999) (logical formula) or (Halber, 2001) (linear types).
2. A "plain" TAG can be either hand-crafted (e.g. (XTAG Research Group, 2001)) or acquired automatically from a TreeBank (e.g. (Chiang, 2000)).
3. For example (Chen, Bangalore and Vijay-Shanker, 1999) report up 98.4% accuracy with an average of 4.8 supertags per word, using the hand-crafted Xtag grammar for English, comprising approx. 600 trees.

 Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+6), pp. 163–168. Universitá di Venezia.

deteriorates with a large POS tagset). To remedy this problem, F. Toussenel and A. Nasr, following a proposal from (Chen, 2001), are associating a "feature vector" to each supertag i.e. a set of of features which capture the salient characteristics of a SuperTag. The idea is then to train a supertagger on each element of the vector in order to reduce sparse data problems. (Chen, 2001) proposes features such as *Part of speech, Subcategorization....By-passive, Wh movement, Topicalized*, but does not detail how to obtain these features from elementary trees [4]. As we discuss below, this kind of features can be organized and extracted from Supertags in a systematic way.

2.2. Text generation

Another application for featurization is text generation. For example, (Bangalore and Rambow, 2000) discuss featurization for text generation as a way to *improve error analysis and allow for better stochastic Supertag assignments models*. For lack of space we refer to their paper page 39 for a list of features they use. [5] Also for text generation, (Danlos, 2000) uses the G-TAG formalism, and the notion of *T-features* which can take the value + or - to identify non-canonical syntactic constructions (i.e. to refer to elementary trees). For instance, if one considers the transitive family, the binary T-features *T-passive, T-without-arg-2* are used to distinguish the salient features of four distinct elementary trees in that family. A given feature structure will then yield a node in a derivation tree. From that derivation tree, a sentence exhibiting the correct syntactic phenomena will be generated

We do not discuss into details the other numerous potential applications of featurizing a TAG, but still mention the use of featurization for sentence planning and query-answer applications (Stone and Doran, 1997), for easy search of annotated corpora (the idea being that it is easier, esp. for non TAG experts, to search a feature such as *relativized object=+* in a corpus where each word is annotated with a feature structure representing its main syntactic characteristic, rather than making queries on the shape of Supertags identifying relativized-objects) (Kinyon, 2000). Finally, featurizing a TAG is a way to achieve framework independence. This can be used to share resources between lexicons, to compile TAGs into different frameworks, and more generally to define a notion of *syntactic interlingua*.

However, to achieve such a goal, one must agree among all the featurizations proposed, on what kind of linguistic featurization is desirable and/or achievable.

4. For a complete list of features proposed see his PhD dissertation pp246-247. Another discussion related to training probabilistic models taking into account syntactic transformations, independently of TAGs, may be found in (Eisner, 2001)
5. Since they deal with both probabilistic supertagging, and text generation, we have made the arbitrary choice of classifying their work in this section instead of the previous one.

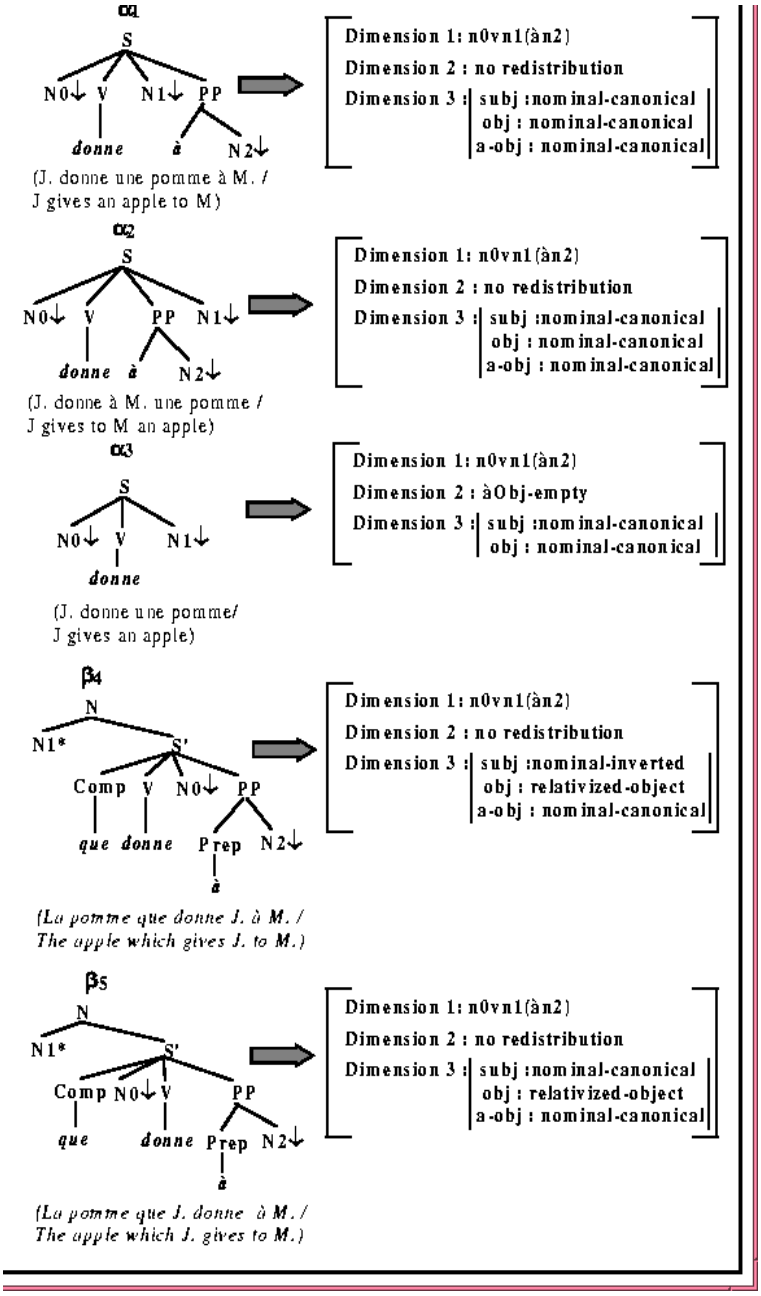

Figure 1: Featurization of elementary trees with a MetaGrammar for French: each elementary tree is associated to a feature structure which contains the main syntactic characteristics of the tree (i.e. the final classes of the hierarchy which allowed the generation of that tree).

3. Featurizing with a MetaGrammar

(Candito, 1996) has originally developed in LISP a tool to semi-automatically generate a 5000-tree wide coverage TAG for French (Abeille, Candito and Kinyon, 1999) as well as a large grammar for Italian (Candito, 1999). The idea is to use an additional layer of linguistic description called the MetaGrammar (MG), which imposes a general organization for syntactic information in a three-dimensional hierarchy :

- Dimension 1: initial subcategorization

- Dimension 2: valency alternations and redistribution of functions
- Dimension 3: surface realization of arguments.

Each terminal class in dimension 1 describes a possible initial subcategorization (i.e. a tree family). Each terminal class in dimension 2 describes a list of ordered redistributions of functions (e.g. it allows to add an argument for causatives). Each terminal class in dimension 3 represents the surface realization of a surface function (ex : pronominalized, wh-extracted etc.). Each class in the hierarchy corresponds to the partial description of a tree (Rogers and Vijay-Shanker, 1994). An elementary tree is generated by inheriting from exactly one terminal class from dimension 1, one terminal class from dimension 2, and n terminal classes from dimension 3 (where n is the number of arguments of the elementary tree being generated). Since the hierarchy summarizes the main linguistic pieces of information that will be used to generate a given elementary tree, when using such a MetaGrammar to generate a TAG, it is possible to keep track of which terminal classes each elementary tree inherits from. Therefore, such a TAG can easily be "featurized" as can be seen on figure 1. This implementation was not included in Candito's tool, but (Crabbe and Gaiffe, 2002) recently reimplemented a MetaGrammar generator in Java, which incorporates such a featurization facility. In addition, (Crabbe and Gaiffe, 2002)'s tool offers flexibility in the sense that one or more features may be associated to each class in the hierarchy, and that the feature(s) associated to a given class may be the name of that class, but may also be any other feature deemed appropriate. This flexibility has obvious advantages, but it may also prove to be an inconvenient in the sense that there is no constraints based on linguistic principles. For example, their implementation does not enforce a three-dimensional hierarchy. It is up to the MetaGrammar writer to make sure that the hierarchy is linguistically adequate. Nonetheless, it is a very useful tool to automatically featurize a TAG.

4. Featurizing with MetaRules

The idea of MetaRules is presented in (Becker, 2000). A MetaRule takes as input an elementary tree and outputs a new, and generally more complex, elementary tree. Therefore, in order to create a TAG, one can start from one *canonical* elementary tree for each subcategorization frame and a finite number of MetaRules which model syntactic transformations (e.g. passive, wh-questions etc) and automatically generate a full-size grammar. [6] Within the Xtag project (Prolo, 2002) was the first one to implement a large TAG grammar for English based on MetaRules and to tackle with the numerous practical implementation issues left unsolved. [7] This grammar consists of 1008 elementary trees and was generated from 57 canonical trees and 21 MetaRules within the Xtag project. It is quite straightforward, when the grammar is generated with MetaRules, to keep track of which MetaRules were applied to generate a given elementary tree. The main characteristic of an elementary tree can therefore be represented in a feature structure with one feature taking as value a subcategorization frame, and 21 binary features (one for each metarule) having the value + if the metarule was applied to generate that tree, and the feature - if it was not. Figure 2 illustrates such a featurization. So this is a second possibility to achieve the automatic featurization of a TAG.

5. Featurizing a plain grammar

To featurize a *plain* grammar, hand-crafted or automatically acquired, one can start from a finite set of features that are judged linguistically interesting (e.g. such as those suggested by (Chen, 2001). One can then perform the tedious task of manually examining each elementary tree in the TAG to assign a value to each of those features for each of these trees (possibly with the help of some degree of automation e.g. macros, pearl scripts etc...). However, ensuring constitency with such a hand-featurization does not seem realistic : ideally we would want a given feature structure to correspond to a unique elementary tree. At the very least, we would want a consistent featurization of all the trees. For example, having a *subcategory* feature for the trees in some family but not for all families would be problematic. Similarly, having a *relativized* feature for only some of the trees exhibiting a relativized argument would be equally unsatisfactory. This type of inconsistency could easily arise though esp.

6. Since we examine **the** one practical existing implementation of MetaRules, we do not take position on the theoretical status of *syntactic transformations* nor on the best way to ensure the finiteness of the generation process (i.e. via finite closure or ordering of the rules etc.) We refer to (Becker, 2000), or (Evans, Gazdar and Weir, 2000) for a slightly passionate discussion on this issue.
7. Unfortunately, apart from (Prolo, 2002), no detailed description of this implementation has been published to this day.

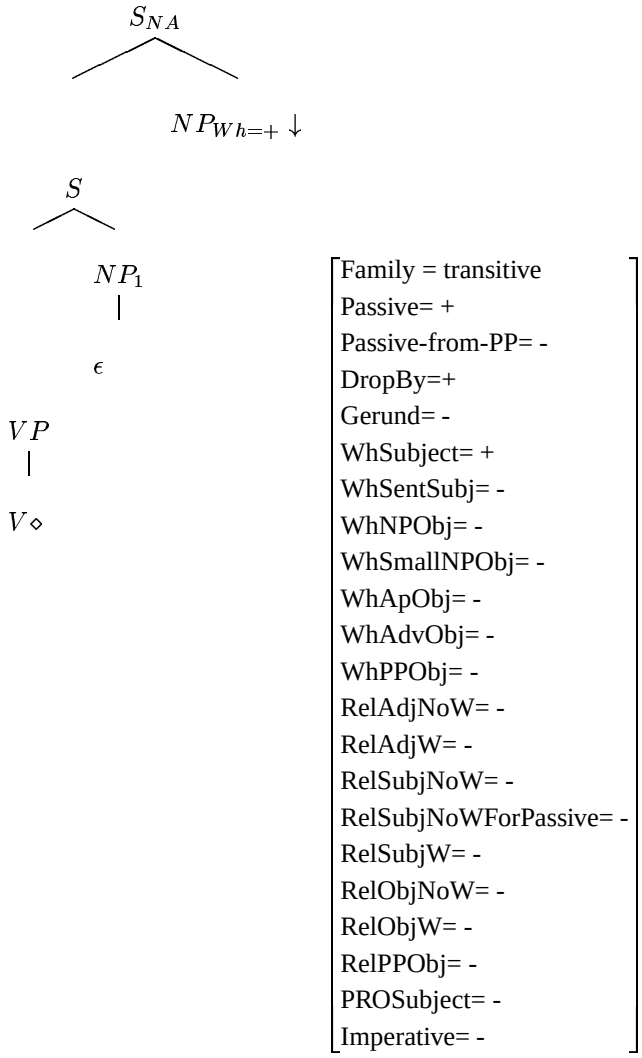

Figure 2: Featurization with MetaRules: elementary tree for *What was eaten* and its associated feature structure (i.e. MetaRules applied to generate the tree are marked +)

when the plain grammar is augmented : when new trees are added to the grammar, manual featurization obliges one to go over the feature structure associated to each new tree in the grammar. Therefore, automatic featurization is higly desirable even in the case of a hand-crafted TAG : given a TAG, a program needs to systematically extract the salient linguistic information from each elementary tree. Note that this amounts to extracting a higher level of syntactic information from the grammar (i.e. a set of MetaRules or the hierarchy nodes of a MetaGrammar). The question remains as to whether it is preferable to extract a MetaGrammar (yielding non binary features) or MetaRules (yielding, at least with the existing implementation, binary features).

It is generally believed that the MetaGrammar approach is preferable because it allows *real* featurization, instead of just a feature vector. This is highly debatable and may be simply due to the fact that the linguistic basis of the MetaGrammar approach have been more cleanly defined i.e. the MG incorporate the notion of syntactic function, and the distinction between subcategorization, valency alternations and realization of arguments.

For Metarules, the notion of initial subcategorization is de-facto captured since a meta-rule grammar is initially generated from a set of elementary trees. In addition to that, it seems highly desirable and feasible to incorporate the notions of syntactic function, valency distribution and realization of arguments into the notion of metarules.

This amounts to formulating metarules in terms of syntactic functions and separating two types of MetaRules :

- Alternation metarules (ordered)
- Realization metarules (unordered)

Moreover, in practice, Metarules are more compact than MetaGrammars for developing *real* grammars (but this point is beyond the scope of this work). The bottom line is that MetaRules and MetaGrammars are not necessarily as different as they are thought to be, and that automatically extracting an abstract level of syntactic representation from a plain TAG yields in fact a hybrid representation that is between the notion of MG and that of MetaRules.

6. Conclusion

We have (briefly) discussed diverse applications for which extracting the salient linguistic characteristics of a TAG via featurization is interesting. We have shown that existing TAG implementations based on MetaGrammar and MetaRules allow automatic featurization. We have argued that automatic featurization is preferable even in the case of *plain* grammars, whether hand-crafted or automatically acquired; and that a MetaGrammar approach is not necessarily preferable to a MetaRule approach. We also would have liked to present the partial implementation which we have done on the PennTreebank to acquire a hybrid level of syntactic representation, mixing MetaGrammar and MetaRules characteristics but we will have to wait for a more suitable presentation format.[8] From this abstract level of syntactic representation, one can generate a TAG, but also potentially grammars in another format (such as LFG, HPSG, dependency grammars etc...). Hence, an abstract level of syntactic representation (MG, MetaRules or Hybrid) can be an interesting syntactic interlingua candidate for compiling one grammar framework into another.

6.1. Acknowlegements

We thank B. Crabbe, C. Prolo, F. Toussenel for their helpful comments on earlier drafts of this paper.

References

Abeille, A., M.H. Candito and A. Kinyon. 1999. FTAG: current status and parsing scheme. In *VEXTAL-99*, Venice.
Bangalore, S and O. Rambow. 2000. Using TAGs, a Tree Model, and a Language Model for Generation. In *TAG+5*, Paris.
Becker, T. 2000. Patterns in metarules for TAG. In Abeille Rambow, editor, *Tree Adjoining Grammars*, CSLI.
Candito, M.H. 1996. A principle-based hierarchical representation of LTAGs. In *COLING-96*, Copenhagen.
Candito, M.H. 1999. *Representation modulaire et parametrable de grammaires electroniques lexicalisees*. Ph.D. thesis, Univ. Paris 7.
Chen, J. 2001. *Towards Efficient Statistical Parsing using Lexicalized Grammatical Information.* Ph.D. thesis, Univ. of Delaware.
Chen, J., S. Bangalore and K. Vijay-Shanker. 1999. New Models for Improving Supertag Disambiguation. In *EACL99*, Bergen.
Chiang, D. 2000. Statistical parsing with an automatically-extracted TAG. In *ACL-00*, Hong-Kong.
Crabbe, B. and B. Gaiffe. 2002. A new Metagrammar Compiler. In *Proc. TAG+6*, Venice.
Danlos, L. 2000. G-TAG: A lexicalized formalism for text generation. In Abeille Rambow, editor, *Tree Adjoining Grammars*, CSLI.
Eisner, J. 2001. *Smoothing a Probabilistic Lexicon via Syntactic Transformations.* Ph.D. thesis, Univ. of Pennsylvania.
Evans, R., G. Gazdar and D. Weir. 2000. Lexical rules are just lexical rules. In Abeille Rambow, editor, *Tree Adjoining Grammars*, CSLI.
Halber, A. 2001. *Strategie d'analyse pour la comprehension de la parole: une approche TAG*. Ph.D. thesis, ENST-Paris.
Kallmeyer, L. 1999. *Tree Description Grammars and Underspecified Representations.* Ph.D. thesis, Univ. of Tubingen.
Kinyon, A. 2000. Hypertags. In *COLING-00*, Sarrebrucken.
Kinyon, A. and C. Prolo. 2002. Identifying verb arguments and their syntactic function in the Penn Treebank. In *LREC 02*, Las Palmas.
Prolo, C. 2002. Generating the Xtag English grammar using metarules. In *Proc. TAG+6*, Venice.
Rogers, J. and K. Vijay-Shanker. 1994. Obtaining trees from their description: an application to TAGS. In *Computational Intelligence 10:4*.
Srinivas, B. 1997. *Complexity of lexical descriptions and its relevance for partial parsing.* Ph.D. thesis, Univ. of Pennsylvania.
Stone, M. and C. Doran. 1997. Sentence Planning as description using TAG. In *ACL97*, Madrid.
XTAG Research Group. 2001. A Lexicalized Tree Adjoining Grammar for English. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.

8. Meanwhile, we refer the reader to (Kinyon and Prolo, 2002) for an incomplete overview of how tree families are extracted from the PennTreebank with an interlingua appraoch.