How to solve some failures of LTAG

Sylvain KAHANE
LaTTiCe – TALaNa (Université Paris 7)
sk@ccr.jussieu.fr

Abstract.

The paper presents a lexicalized dependency grammar which solves some failures of Lexicalized TAGs, such as the combinatorial explosion of the number of elementary trees and the non adequacy for the analysis of some constructions.

Introduction

Wide coverage grammars for natural languages have been developed in Lexicalized TAG (cf. Abeillé 1991, Candito 1999 for French and Paroubek *et al.* 1992, XTAG 1995 for English). These implementations have brought to the fore some failures of the formalism for natural language description which cannot be solved without adopting a descriptively more powerful formalism. These failures concern most of lexicalized grammars, including Categorial Grammars (CG). In this paper, we will present some of these failures and propose solutions in a lexicalized dependency grammar based on Nasr 1995, 1996.

1. Lexicalized grammars

An LTAG is a particular case of **lexicalized grammar** (LG). A LG is a formal grammar that has the form of a lexicon: each lexical unit is associated to a set of elementary structures. The grammar has an operation of combination[1] and each sentence (= a string of word) can be associated to set of structures obtained by combinations of elementary structures associated to the words of the sentence.

Formally, a LG is a 5-uple $G = < \mathcal{L}, S, S_F, \varphi, c >$ where:

- $\mathcal{L}$ is the lexicon;
- S is the set of structures; it is an infinite set but it must be finitely defined;
- S_F is the subset of S of final structures;
- φ is a many-to-many map from $\mathcal{L}$ to S;
- c is the operation of combination of structures; it is a many-to-many map from $S \times S$ to S.[2] Below $c(\alpha,\beta)$ will be noted $\alpha.\beta$.

The operation c induces an operation c^* from S^* to S which associates to a sequence of structures of S all the structures of S obtained by combination of these structures. For instance, $c^*(\alpha,\beta,\gamma)$ is all the structures obtained by the combinations $(\alpha.\beta).\gamma$ and $\alpha.(\beta.\gamma)$. The grammar G defines a correspondence (= many-to-many map) φ^* between $\mathcal{L}^*$ and S_F: a sentence $u = x_1 x_2 \ldots x_n$ in $\mathcal{L}^*$ and a structure S in S_F are in correspondence if for each word x_i there is a structure $S_i = \varphi(x_i)$ such that $c^*(S_1, S_2, \ldots, S_n) = S$.

---

[1] Most of formalisms consider several operations of combinations (e.g. substitution and adjoining in TAG), but we can suppose that there is only one, which is the union of all of them.
[2] We do not exclude that two structures can combine in several ways.

We will now present an LG adapted from Nasr 1995, 1996, which we call Lexicalized Dependency Grammar (LDG). The set of structures S of LDG is a set of dependency trees (Tesnière 1959, Mel'čuk 1988); nodes are labeled by a lexical unit, its part of speech and some grammatical features (not considered here), while branches are labeled by a syntactic relation and a weight (see below). Moreover each label contains a feature `type` with value 0 (= white) or 1 (= black) such that $0 \cup 0 = 0$, $0 \cup 1 = 1$ and $1 \cup 1 =$ `failure`. S_F is the subset of dependency trees in S whose all nodes and branches are black, that is have the value `type:1`. The feature `type` ensures that each element is build one and only one time: black elements can be considered as element which are build and white elements, as requests.

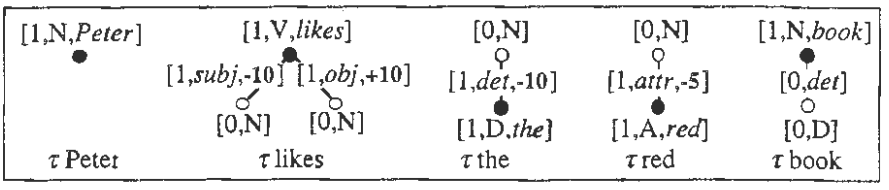

Figure 1. Elementary trees

In the plain case, elements of S combine by unification of one node. In some cases, several nodes and branches can unify (e.g., the combination of the tree of *book* with the tree of its determiner *the*, Fig.2). The feature `type` allows a black element to unify only with a white element.

A sentence u **corresponds** to a tree T of S_F if:

- the nodes of T are labeled by the words of u and **correspond one-to-one** to them;
- the product structure T×u, that is the tree T with the linear order on the nodes induced by u, is a **projective** ordered tree (no arcs cross each other and no arc covers the root);
- the **local order constraints** given by the weights on the branches are respected: the sign of the weight (- or +) indicates if the dependent is before or after the governor and the absolute value of the weight indicates the relative distance between the dependent and the governor.

Fig. 2 shows the dependency tree resulting from the combination of the elementary trees of Fig. 1 and the correspondence between this tree and the sentence *Peter likes the red book*.

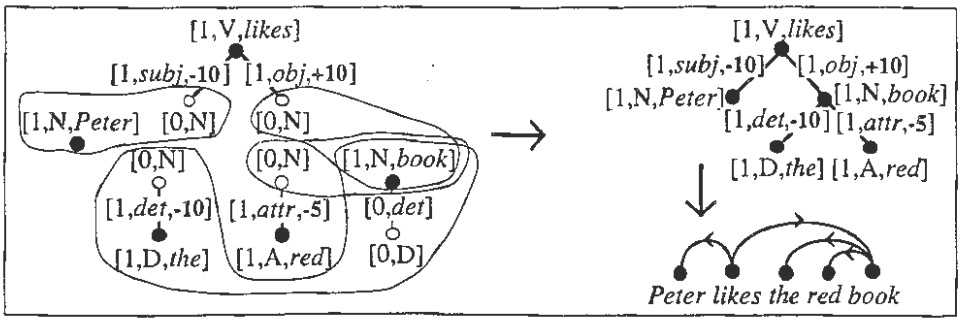

Figure 2. Combination

2. Avoid the combinatorial explosion of the number of elementary trees

The first failure of LTAGs is certainly the combinatorial explosion of the number of elementary trees associated to a given lexical unit. Due to the fact that for each non-canonical position of an

argument (deletion, topicalization, inversion, relativization, cliticization, raising, heavy shift...) a different tree is necessary and due to the crossing with the different arguments, several hundred of elementary trees can correspond to a same lexical unit. Tools have been proposed to write grammars in formalisms which avoid redundancies and allow to generate an LTAG (cf. Vijay-Shanker & Schabes 1992, Candito 1996, 1999). But such formalism—called a **metagrammar** in Candito 1999—cannot be used directly as a grammar and must be compiled into a LTAG before using. And due to the great number of elementary trees, LTAG parsers are not very efficient and consume a lot of space memory.[3] Our proposition consists to propose a lexicalized grammar which has more or less the property of a metagrammar, but which can be used directly as an LTAG.

We claim that the number of elementary trees associated to a lexical unit depends on two factors:

1) the repartition of the linguistic information;

2) the expressiveness of S and the powerfulness of c.

We will now study some examples and propose solutions with our LDG.

Attribute and predicative adjectives. In LTAG, adjectives receive two different elementary structures for their attribute and predicative uses. Compare *the red book* and *the book is red*. The LTAG's elementary tree of the attributive *red* has a nominal foot node in order to adjoin on a noun (here *book*), while the LTAG's elementary tree of the predicative *red* has a nominal substitution node (occupied here by *book*) and a verbal node where the copulative verb will adjoin. But the particular behavior of predicative adjective can be attributed to the copulative verbs rather than to the adjective and a same elementary tree should be attributed to attributive and predicative adjectives. But the TAG formalism is not powerful enough for that. Our LDG can be enriched to solve this problem. We consider a new type of branches, called **quasi-dependency**, with a feature `+quasi`. Quasi-dependencies do not intervene in the tree hierarchy nor in the linearization (they do not bear a weight), but they can unify with a true dependency (the result is still a quasi-dependency). The elementary tree τred (Fig.1), which is used for attributive constructions (Fig. 2), can also be used for predicative constructions. In this case, the *attr* dependency adjoins with a quasi-dependency of the elementary tree of the copula (Fig. 3). In other words, we have given the copula the power to absorb this dependency and to give another syntactic governor to *red* than the noun governing it in its elementary tree. The problem has been solved by adopting a different repartition of linguistic information (properties of predicative constructions are attached to the adjective to the copula, rather than to the adjective as in LTAG), which was made possible by an enrichment of the formalism.[4]

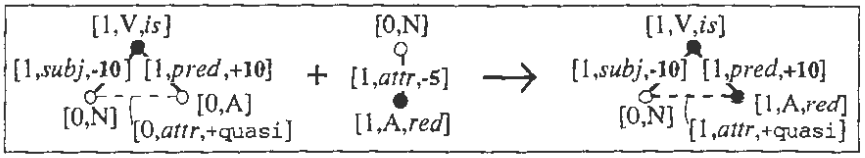

Figure 3. Derivation of [*The book*] *is red*

[3] Parsing algorithm for LTAG have time complexity in $C.|G|^2.n^7$ and space complexity in $C.|G|.n^3$, where $|G|$ is the size of G, that is the number of elementary structures.

[4] The problem can also be solved in CG: the noun *book* will receive the category N and the adjective *red* the category N/N, in order to adjoin on the noun. Then the copula receives the category N\S/(N/N). Nevertheless, CG presents some failures; in paticular, CG has not a convenient treatment of adjoining. For instance, if we want to specify that a noun must have a determiner we will give it the category D\N, but, in this case, *red* must receive the category (D\N)/(D\N). And if several categories are considered for nouns, several categories must be considered for *red*. Another point: at first view, CG is not exactly a lexicalized grammar in the sense considered here, because the combination of categories does not build. But a structure can be derived from the reduction process or categories can be enriched with lambda terms whose combination gives a semantic structure.

Non-canonical position. In LTAG, all the arguments are positioned in the elementary tree of their governor. But the particular behavior of some elements (wh-words, clitics...) might be attributed to them rather than to their governor. And again the TAG formalism is not sufficiently powerful for that.

Our LDG can be enriched to solve this problem. We consider a new type of feature value, called **priority value**: rather to unify with another value, a priority value replaces it. Fig. 4 proposes a solution for clitics in French. Object clitic *le* is positioned before the verb and the relative order of clitics is very constrained (roughly *se* < *le* < *lui* < *en* < *y*). Therefore, the clitic *le* will receive an elementary tree with a white *obj* governor dependency bearing a priority weight of –4; consequently, the clitic *le* can only combine with an *obj* request and its priority weight value will ensure its correct positioning. In our figures, priority values are underlined.

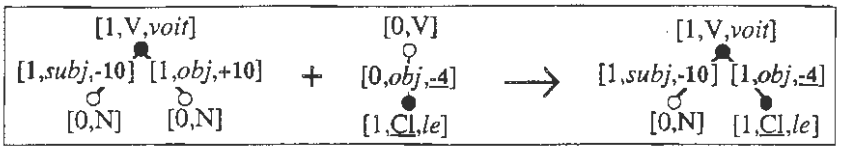

Figure 4. Derivation of Fr. [*Pierre*] ***le voit*** 'Peter sees it' (first proposal)

Non projective constructions. Our first proposal for clitics operates only for projective case, that is when the clitic is on the word that subcategorizes it. We will propose here a solution for **clitic climbing** in French:

(1) *Pierre l'a vu*, lit. Peter LE has seen 'Peter has seen it'

(2) *Pierre en aime la fin*, lit. Peter EN likes the end, 'Peter likes the end of it'

Case (1) is solved in LTAG by adjoining the auxiliary verb *a* 'has' on the past participle *vu* 'seen' (Abeillé 1991). It is not satisfactory because the auxiliary is the syntactic head of the clause; for instance, it receives the negation *ne...pas*: *Pierre ne l'a pas vu*, Peter NE LE has not seen, 'Peter has not seen it'. This last sentence cannot be satisfactorily derived in LTAG, because the clitic *ne*, which is borne by the auxiliary, cannot adjoin on it because of the clitic *le*, which is on the tree of the past participle. The case of (2) is even more problematic: the only way to solve it is to use set-local multi-component TAG (Bleam 1994).

Our solution is inspired from Hudson 2000 and can be compared to the Slash analysis: the clitic is lifted from its **syntactic governor** (the word which subcategorizes it) to its **linear governor** (the word on which it positions). As the dependencies are used for the linearization, the clitic must depends on its linear governor by a true dependency (with a weight), while the dependency with its syntactic governor (in the elementary tree of its syntactic governor) must become a quasi-dependency. For these reasons, the elementary structure of a clitic has a dependency labeled ***aff(ix)*** linked to its linear governor, which ensures its good linearization, and a quasi-dependency linked to its syntactic governor, which must unify with the request of its syntactic governor (Fig. 5).[5] The most difficult problem is to ensure that the clitic climbs on the good node. The lifting is controlled by a bubble, labeled β, containing both syntactic and linear governors of the clitic. We assume that a dependency on a node of a β bubble will be contained in the β bubble if and only if it is labeled iβ. Therefore, when the clitic's elementary tree τ*le* combines with the auxiliary verb's elementary tree τ*a*, the *aux* dependency of τ*a*, which is labeled iβ, must be contained in the bubble β. Moreover the

[5] Note the particular treatment of the past participle: it has a subject but this subject is linked by a quasi-dependency. This quasi-dependency unifies with the quasi-dependency of the auxiliary elementary tree. The "subject" of the past participle cannot be realized (and linearized) without being linked to the tree by a true dependency.

tree *tle* indicates that the linear governor of the clitic must be a finite or infinite verb. As the syntactic governor is not finite or infinite, the clitic climbing is needed.

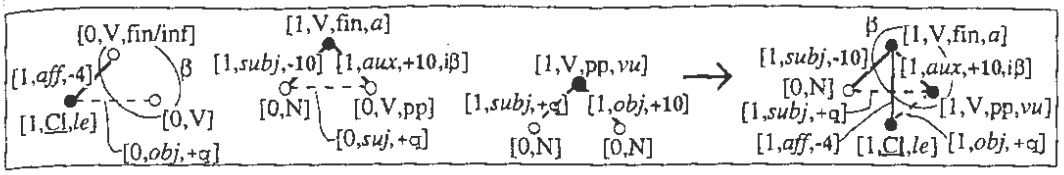

Figure 5. Clitic climbing (derivation of (1))

Even when there is no climbing, the same elementary tree can be used for the clitic: in this case, the two nodes of the β bubble unify and there is a dependency and a quasi-dependency between the clitic and its (syntactic and linear) governor (Fig. 6).

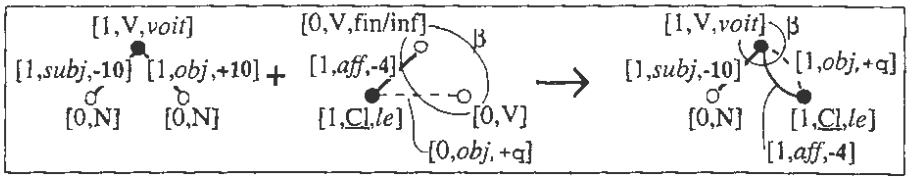

Figure 6. Derivation of Fr. [*Pierre*] ***le voit*** 'Peter sees it' (second proposal)

Let us come back to the problem of the negation *ne...pas*, which cannot be solved satisfactorily in TAG. The negation simply adjoins to the finite verb, *ne* with a weight -5 and *pas* with a weight +2.

Kahane 2000 proposes a similar solution for extractions.

3. Syntax and semantics

One of the main interest of LTAG is that the derivation tree can be interpreted as a semantic graph (= predicate-argument structures) (Candito & Kahane 1998a). To allow such an interpretation, some principles are required: the lexical nodes of an elementary tree must correspond to exactly one semantic unit (Abeillé 1991) and the non-lexical leafs of an elementary tree corresponds one-to-one to the arguments of this semantic unit (predicate-argument co-occurrence principle). But a strict application of this principle is too strong: for instance, it forbids that a syntactic element such as a copulative verb or a complementizer anchors its own tree. In the same way, it forbids that a lexical unit combines with a syntactic argument which is not a semantic argument such as the subject of a raising verb (such has *Peter* with *seems* in *Peter seems to be sleeping*). Such principles forbid also having a separate tree for the copulative verb, which is semantically empty.

Our solution consists in establishing the semantic connection, as in LTAG, while keeping the syntactic connections. In this case, it becomes necessary to indicate explicitly the semantic connection. For this reason, each node receives a `sem` feature, whose value is the semanteme corresponding to the word, and an `arg` feature, whose value is the list of the semantemes of its arguments. The elements of this list are equal to the `sem` values of the argument, which is indicated in the elementary tree by shared values.

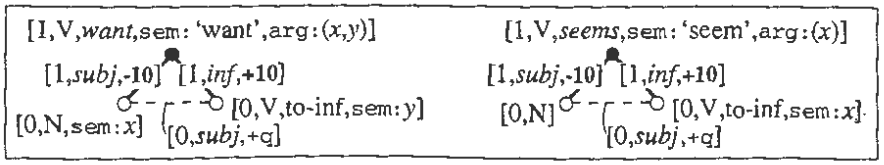

Figure 7. Control verb and raising verb

Fig. 7 gives us the elementary trees of the control verb *want* and of the raising verb *seem*; they have the same syntactic trees (in particular both have a syntactic subject) but they differ semantically: only the control verb has its syntactic subject as semantic argument. Moreover, our formalism allows recuperating directly the semantic dependencies even when there is a cycle (Fig. 8), which TAG cannot allow us.

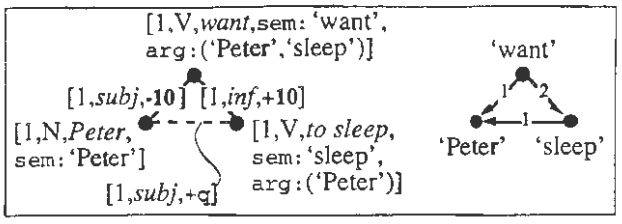

Figure 8. The structure and the corresponding semantic graph of *Peter wants to sleep*

4. Conclusion

Our conclusion is that the TAG formalism is not powerful enough to reach the objectives of computational and linguistic adequacies required to it. Nevertheless, it is possible to develop near formalisms which reach these goals, as well as they keep its advantages, such as lexicalization, simplicity of the operation of combination or readability of the elementary structures.

5. References

ABEILLÉ Anne (1991) : *Une grammaire lexicalisée d'Arbres Adjoints pour le français*, Thèse de Doctorat, Univ. Paris 7.

BLEAM Tonia, (1994) : "Clitic Climbing and the Power of Tree Adjoining Grammar", in *Symposium on TAG*, Paris. To appear in Abeillé A. & Rambow O., *Tree Adjoining Grammar*, CSLI.

CANDITO Marie-Hélène (1996) : "A Principled-based Hierarchical Representation of LTAG", *COLING'96*, Copenhague, pp. 194-99.

CANDITO Marie-Hélène (1999) : *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien*, Thèse de Doctorat, Univ. Paris 7.

CANDITO Marie-Hélène & KAHANE Sylvain (1998a) : "Can the TAG Derivation Tree represent a Semantic Graph ? An Answer in the Light of Meaning-Text Theory", *TAG+4*, Philadelphie, pp. 25-28.

CANDITO Marie-Hélène & KAHANE Sylvain (1998b): "Defining DTG Derivations to get Semantic Graphs", *TAG+4*, Philadelphie, pp. 25-28.

GERDES Kim (1998) : *Le cas allemand en TAG*, Mémoire de DEA, Univ. Paris 7.

HUDSON Richard (2000) : "Discontinuity", *Special Issue on Dependency Grammar, T.A.L, 41:1*, , Paris, 38p.

KAHANE Sylvain (1997) : "Bubble Trees and Syntactic Representations", in Becker & Krieger (eds), *Proc. MOL'5*, Saarbrücken : DFKI, 70-76.

KAHANE Sylvain (2000) : "Une grammaire de dépendance à bulles pour traiter l'extraction", *Special Issue on Dependency Grammar, T.A.L, 41:1*, Paris, 30p.

MEL'CUK Igor (1988) : *Dependency Syntax: Theory and Practice*, NY : State Univ. of NY Press.

NASR Alexis (1995) : "A Formalism and a Parser for Lexicalised Dependency Grammars", *4th Int. Workshop on Parsing Technologies*, State Univ. of NY Press.

NASR Alexis (1996) : *Un modèle de reformulation automatique fondé sur la Théorie Sens-Texte – Application aux langues contrôlées*, Thèse de Doctorat, Univ. Paris 7.

PAROUBEK Patrick, SCHABES Yves & JOSHI Aravind K. (1992) : "XTAG; a graphical Workbench for developing TAGs", *ANLP*, Trento, 223-27.

RAMBOW Owen (1994) : *Formal and Computational Aspects of Natural Language Syntax*, PhD Thesis, Univ. Of Pennsylvania, Philadelphia.

TESNIÈRE Lucien (1959) : *Eléments de syntaxe structurale*, Paris : Kliencksieck.

XTAG Research Group (1995) : *A Lexicalized TAG for English*, Technical Report IRCS 95-03, Univ. of Pennsylvania, (updated version on the web).