# Modular and Hierarchical Organization of Lexicalized Grammars expressible with Tree Descriptions

Guy Perrier

# Modular and Hierarchical Organization of Lexicalized Grammars expressible with Tree Descriptions

**Guy Perrier**
LORIA - CNRS
Campus Scientifique - BP 239
54506 Vandœuvre-lès-Nancy Cedex -France
e-mail: perrier@loria.fr

## Abstract

We propose a formal framework for a modular and hierarchical organization of lexicalized grammars which are expressible with tree descriptions.
The typical syntactic constructions of a grammar are expressed with tree descriptions embedded in modules which are organized into a hierarchy by a multiple inheritance relation.
The syntactic constructions that constitute the entries of the lexicon are built automatically by crossing the terminal modules of this hierarchy under two kinds of constraints: negative constraints which are expressed by disjunctive inheritance and which forbid crossing some modules and positive constraints which are expressed by co-occurrence relations between modules and which force the crossing of other modules.
The lexical modules that result from this crossing process are linked to the words of the language in the lexicon by a mechanism that is based on feature structures. Every module of the grammar is associated with a feature structure or a disjunction of feature structures which describes the morpho-syntactic profile of the words of the language that can anchor the syntactic construction described by the module. Unification is used for generating the profiles of the lexical modules in parallel with the processes of module inheritance and crossing.

## Introduction

The organization of wide coverage lexicalized grammars requires mechanisms for factorizing information and capturing generalization in a crucial way both from a linguistic and a computational point of view. This need of factorization appears as much at an abstract representation level as at the level of the concrete implementation of grammars and lexicons. In this paper, we focus on the abstract level of representation.
Feature-based grammars answer the question by exploiting their basic mechanism of unification between feature structures: they are organized into a hierarchy of more or less general syntactic constructions represented by feature patterns. The relation of subsumption between feature structures is used as an inheritance relation for structuring the hierarchy. The management of the inheritance graph can be controlled by assigning types to feature structures (Carpenter, 1992). Finally, inheritance is often combined with lexical rules for expressing transformations of syntactic structures from canonical configurations. The most famous example of such an organization is given by HPSG (Pollard and Sag, 1994).
The same ideas were extended to tree-based grammars. In particular, (Evans et al., 1995) propose to use the lexical knowledge representation language DATR (Evans and Gazdar, 1996) for representing TAGs. With DATR, all syntactic information, even the topology of trees and the lexical rules for syntax revision, is encoded into a hierarchy of feature structures. This system is also used in the LexSys project (Carroll et al., 1998) for representing Lexicalized Description Tree Grammars (LDTG) (Rambow et al., 1995) in a compact way.
The notion of *tree description* opens another direction for the modular representation of grammars. This notion was introduced by (Vijay-Shanker, 1992) for representing TAG trees in a monotone way and it was formalized logically by (Rogers and Vijay-Shanker, 1994). A tree description is a logical formula that describes a tree specification by asserting various relations between nodes: parenthood, domination, precedence, equality and inequality. This notion allows a modular and hierarchical organization of grammars.
This idea is exploited for the representation of the English and Chinese grammars in the XTAG system (XTAG, 2001) and it forms the foundation for a modular organization of French and Italian TAGs (Candito, 1999). In what M.-H. Candito calls a meta-grammar, the syntactic constructions of the language are represented by tree descriptions and are organized into a hierarchy of modules by a multiple inheritance relation. Then, crossing the terminal modules of this hierarchy under some conditions, automatically produces the families of TAG trees which are attached to the entries of a syntactic

lexicon.
The interest of this proposal is to capture linguistic generalizations in a shrewd manner. Its defect is that the presentation of the module hierarchy is not purely declarative: the modules that describe redistributions of syntactic functions are presented in the shape of procedures which revise the values of some features. This entails non monotonicity of the inheritance relation; in this way, when a module inherits from several other modules, the inheritance order from these modules becomes significant; this is the same for the order in which several terminal modules are crossed to produce a unique module. Another defect is that the crossing mechanism between terminal modules is not defined in a uniform way from abstract principles but depends on the concrete organization of the grammar into several dimensions (three for verbs) which are not treated in the same manner. It also depends on constraints of positive and negative co-occurrence between modules.
Starting from the idea of (Candito, 1999), we propose a formal framework for the organization of lexicalized grammars into a hierarchy of syntactic modules which apply to all formalisms that are expressible with tree descriptions. The main originality of this proposal with respect to (Candito, 1999) lies in three points:

- The representation of modules is purely declarative and the inheritance relation is perfectly monotone.
- The crossing mechanism between terminal modules to produce *lexical modules* is uniform; it is controlled by two kinds of constraints: negative constraints which are expressed by disjunctive inheritance and which forbid some crossings and positive constraints which are expressed by co-occurrence relations between modules and which force crossing.
- The mechanism which anchors the lexical modules to the words of the language is based on the association of every syntactic module with a feature structure or possibly a disjunction of feature structures which represent the morphological and syntactic properties of the words of the language which are able to anchor this module.

This framework applies to all lexicalized formal grammars for which the syntactic structures can be represented with tree descriptions. This is the case of TAG and of their extensions such as D-Tree (Substitution) Grammars (Rambow et al., 1995; Rambow et al., 2001). This is also the case of Dependency Grammars provided they are formalized in a lexicalized way (Nasr, 1995). Apparently, Categorial Grammars where syntactic structures are expressed by logical formulas and proofs are far from this framework but (Perrier, 2001) has shown that proofs in a fragment of intuitionistic linear logic, which is the logic kernel of Categorial Grammars, can be expressed as specific models of tree descriptions. As a consequence, Categorial Grammars can be also related with formalisms expressible with tree descriptions.

## 1 A grammar as an inheritance hierarchy of tree descriptions

Intuitively, a modular grammar is structured as a finite set of *syntactic modules* which are linked together by *multiple inheritance*. Every module represents a syntactic construction which can be isolated inside the concerned language. It is defined in the shape of a *tree description* in the sense given by (Rogers and Vijay-Shanker, 1994): the nodes of the description represent syntactic constituents and their set is structured by *domination*, *parenthood*, *precedence* and *equality* relations. The linguistic properties of the constituents are described by means of features labelling the nodes; the set $\mathcal{L}$ of labels used for this purpose depends on the concerned grammatical formalism: for instance, a label can be simply a syntactic category or, in a more sophisticated way, a feature structure.

**Definition 1.1** *A* syntactic module *is a tree description defined in the shape of a classical logic formula by the following grammar:*

$$D ::= D \vee D \mid D \wedge D \mid N > N \mid N >^* N \mid N \prec N \mid N = N \mid N : L$$

*In this grammar,* N *represents any node identifier taken from a set* $\mathcal{N}$ *of node names and* L *represents any label of* $\mathcal{L}$*. The relations* $>$, $>^*$ *and* $\prec$ *represent parenthood, domination and precedence relations between nodes.*

A set of syntactic modules is structured according to an *inheritance order* which is compatible with the contents of the modules: if a module $D_1$ inherits from a module $D_2$, there is a syntactic description $D_1'$ such that $D_1 = D_2 \wedge D_1'$.
Figure 1 gives an example of a hierarchy of modules for a simplified and restricted grammar $G_{verb}$ of verbs in French. Its internal logic must be understood as follows:

- Every verb is provided with the syntactic functions of its arguments in a canonical setting, called *initial functions*; here, we consider only the functions *initial subject* and *initial object*.
- These initial functions must be realized, which is expressed by the modules *init-subj-real* and *init-obj-real*. The realization is performed in the shape of final or intermediate syntactic functions. Here, we consider three final functions: *final subject*, *final object* and *final agent*. The

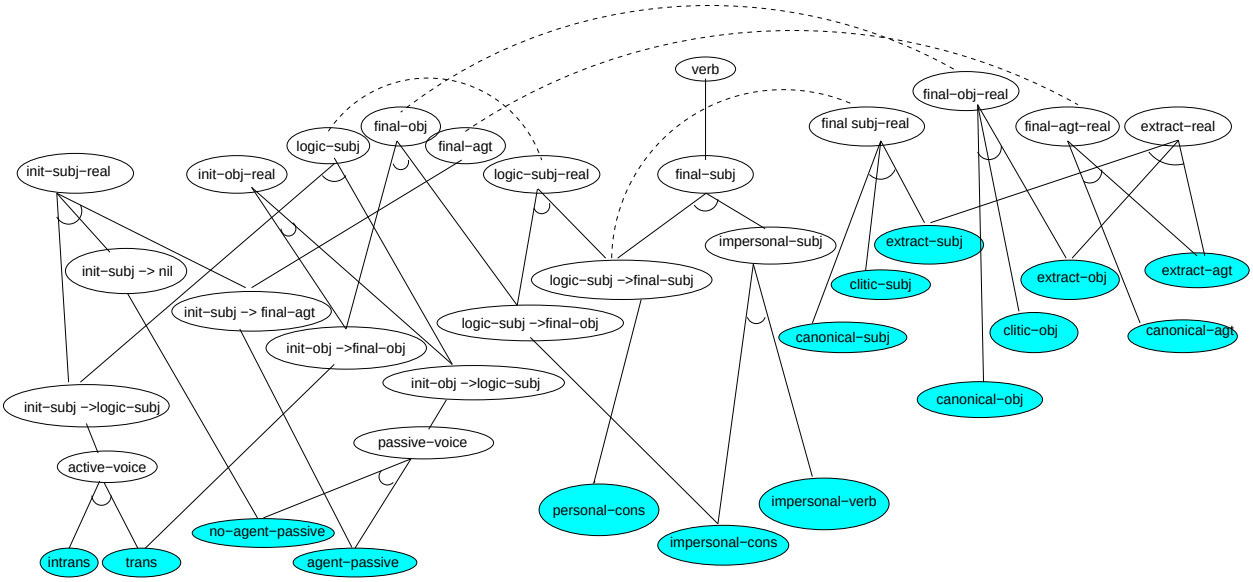


Figure 1: A small modular grammar $G_{verb}$ of verbs in French

requirement of these functions is expressed by the respective modules *final-subj*, *final-obj* and *final-agt*. There is one intermediate function, *logic subject* which is realized in the shape of either a final object in an impersonal construction or a final subject in a personal construction. For instance, the logical subject of the sentences *Un homme arrive (A man is arriving)* and *Il arrive un homme (There is a man arriving)* is *un homme (a man)*. The requirement and the realization of this function are expressed by the respective modules *logic-subj* and *logic-subj-real*.

- Finally, every final function is realized in a canonical, a clitized or an extracted manner[1]. This corresponds to the three modules *final-subj-real*, *final-obj-real* and *final-agt-real*.

If we assume that the grammar $G_{verb}$ aims to generate a lexicon of a Tree Adjoining Grammar, the contents of its modules have the shape of tree descriptions in the sense given by (Vijay-Shanker, 1992). The main characteristic of these descriptions is that the nodes of elementary TAG trees where adjunction is allowed are represented by pairs of quasi-nodes of tree descriptions which are linked together with a domination relation. Figure 2 presents the contents of some modules of the grammar $G_{verb}$ in the shape of tree descriptions devoted to TAG [2]. For instance, the domination relation between the nodes *vmax* and *vanch* in the module *verb* represents an adjoining node in the corresponding TAG tree.
By keeping the architecture described by Figure 1 for the grammar, we have only to change the contents of its modules to obtain a grammar in a different formalism.

## 2 The generation of lexical modules by crossing terminal modules under constraints

From the hierarchy of modules we deduce all grammatical constructions of the associated language by crossing terminal modules, that is modules without sub-modules. Since a module is described by a logical formula, crossing two modules produces a new module which is their logical conjunction. Module crossing is limited by two kinds of constraints:

1. Some modules are declared as *disjunctive modules*, which means that their immediate sub-modules are exclusive and they cannot be crossed directly or indirectly via terminal modules which inherit from them.
2. There is a symmetric binary relation of *co-occurrence* between modules. If two modules are co-occurrent, every terminal module that inherits from the first one, must be crossed with a terminal module which inherits from the second one.

In Figure 1, all non-terminal modules are disjunctive modules and co-occurrence relations are represented

[1] We ignore the subject inversion which depends on particular constructions like interrogations.

[2] If a module inherits from another one, only its specific part is drawn in the corresponding ellipse. Co-indexation of features is represented by integers bracketed with $< >$. Precedence relations between nodes are expressed with arrows.

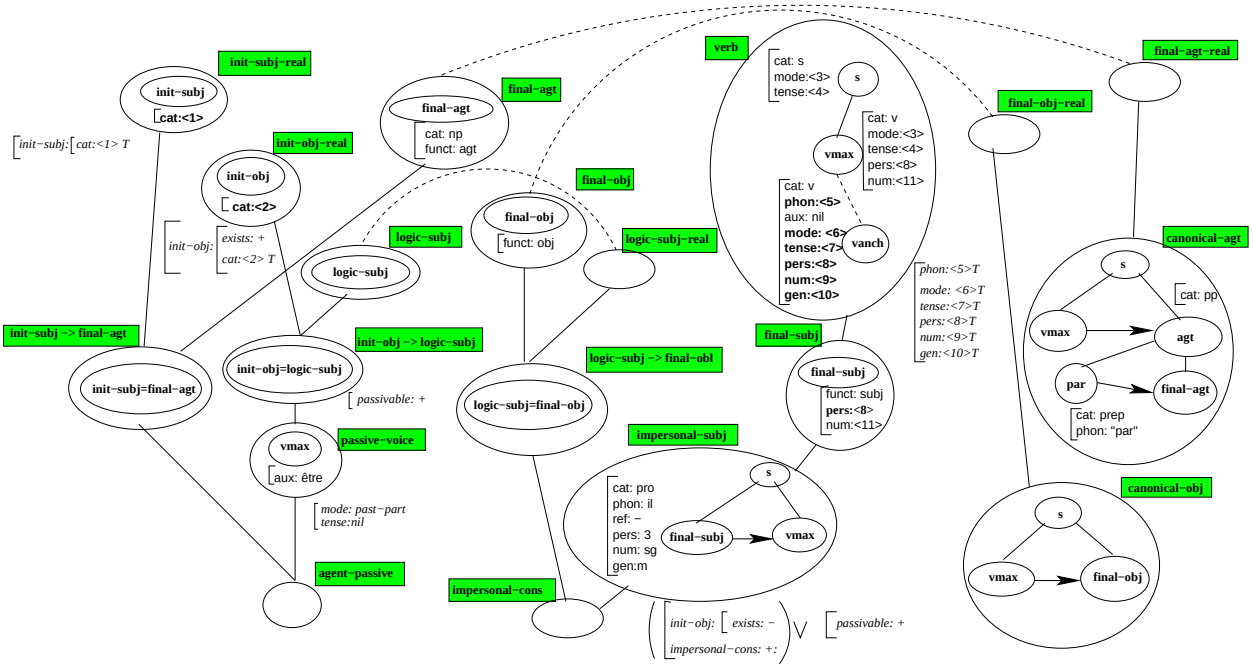


Figure 2: TAG oriented contents of modules for a fragment of $G_{verb}$

by dashed edges. Disjunctive inheritance and co-occurrence act as negative and positive constraints for controlling the process of crossing terminal modules. We have chosen to express these two kinds of constraints in this form for practical reasons but we could replace disjunctive inheritance with a binary relation of *inconsistency* and the co-occurrence relation with *conjunctive inheritance*. The essential is to provide the grammarian with formal tools that are simple enough to use and at the same time expressive enough for representing various choices in the organization of grammars. Let us illustrate this with some examples taken from the grammar $G_{verb}$.
In the grammar $G_{verb}$, the module *impersonal-verb* cannot be crossed with another one and it constitutes a lexical module by itself. If we try to cross it with another terminal module, because of co-occurrence relations, we must cross it with one of the modules *personal-cons* or *impersonal-cons*; since *impersonal-verb* and *personal-cons* inherit from two different immediate sub-modules of the disjunctive module *final-subj*, they cannot be crossed and the same argument works for *impersonal-verb* and *impersonal-cons* with respect to *impersonal-subj*.
The impossibility of using a transitive verb in the impersonal form is expressed by the impossibility of crossing the module *trans* with the module *impersonal-cons* because both inherit from the disjunctive module *final-obj*.
Finally, the possibility of extracting only one argument of the verb at the same time is expressed by the disjunctive module *extract-real*.
By systematically crossing the terminal modules of $G_{verb}$ under the positive and negative co-occurrence constraints, we obtain 31 lexical modules:

- 1 lexical module with a unique component;
- 12 lexical modules with three components;
- 18 lexical modules with four components.

Figure 3 presents an example of a lexical module which results from crossing the terminal modules *agent-passive*, *impersonal-cons* and *canonical-obj*. This lexical module represents the canonical impersonal passive construction with agent which is used in the sentence:

*Il a été mangé un kilo de pommes par Pierre*
*("There was eaten one kilo of apples by Pierre")*

Since the order between the agent and the object is not specified, the lexical module represents two TAG trees.
Now, we are in a position of defining a *modular grammar* formally.

**Definition 2.1** *A modular grammar* G *is a quadruple* $(M_G,\ H_G,\ D_G,\ C_G)$ *such that:*

- $M_G$ *is a finite set of* syntactic modules*;*
- $H_G$ *is an* inheritance *order on* $M_G$ *such that if any module* $D_1$ *inherits from a module* $D_2$*, there exists a tree description* $D'_1$ *with the property:* $D_1 = D_2 \wedge D'_1$*;*

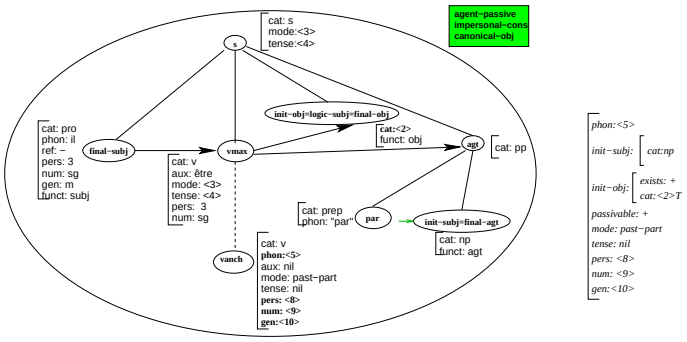


Figure 3: A lexical module produced by the grammar $G_{verb}$

- $D_G$ *is a subset of* $M_G$ *which contains its* disjunctive *modules;*
- $C_G$ *is a symmetric binary relation on* $M_G$ *called* co-occurrence.

The definition of *modular grammar* is indissociable from that of *lexical module.*

**Definition 2.2** *A lexical module* $L$ *of a modular grammar* $G$ *is a conjunction of terminal modules of* $G$ *such that:*

- *two components of* $L$ *do not inherit from two different immediate sub-modules of the same disjunctive module;*
- *if a component of* $L$ *inherits from a module* $D_1$ *which is co-occurrent to another module* $D_2$*, there must be a component of* $L$ *which inherits from* $D_2$*.*

## 3 The lexical anchoring of a modular grammar with feature structures

A modular grammar is not a lexicalized grammar and cannot be used directly for parsing the corresponding language. The lexical modules that are automatically generated from it are not linked to the words of the language. For linking them, we define a mechanism that is based on the notion of *morpho-syntactic profile* of a syntactic module.
Every module of the grammar is associated with its morpho-syntactic profile which summarizes the morphological and syntactic properties which are required for the words of the language to anchor the syntactic construction described by the module. A morpho-syntactic profile has the shape of a feature structure or a disjunction of feature structures. Feature structures are not re-entrant but they are recursive in the sense that a feature value itself can be a feature structure.

**Definition 3.1** *A* morpho-syntactic profile *is a feature structure or a disjunction of feature structures. A* feature structure *is a partial function from a finite set* $\mathcal{A}$ *of feature names to a set of feature values. Any* feature value $V$ *is defined by the grammar:* $V ::= V_0 \mid S \mid V \vee V$ *where* $V_0$ *is any atomic value taken from a countable set* $\mathcal{V}$ *and* $S$ *is any feature structure.*

We extend the usual operations of *unification* and *matching* by taking disjunction into account. Since the notion of feature value includes the notion of profile, we define these operations on feature values. For the sake of uniformity, we consider two constants: $\top$ which unifies with any other feature value and $\bot$ which unifies with no other feature value. These have the following properties: for any feature value $V$, $\top \vee V = \top$ and $\bot \vee V = V$.

**Definition 3.2** *The result of matching a feature value* $U$ *with a feature value* $V$ *is a feature value* $\text{match}(U, V)$ *which is defined recursively according to four cases:*

**$U$ or $V$ is a disjunction:**
$U = U_1 \vee \cdots \vee U_n$ *and* $V = V_1 \vee \cdots \vee V_m$.
*Then* $\text{match}(U, V) = \bigvee_{i=1}^{n} \bigvee_{j=1}^{m} \text{match}(U_i, V_j)$.

**$U$ and $V$ are both feature structures:**
*If a feature is present in* $V$ *and not in* $U$*, then* $\text{match}(U, V) = \bot$. *If a feature* $A$ *is present in* $U$ *and* $V$ *with the values* $X$ *and* $Y$ *and if* $\text{match}(X, Y) = \bot$*, then* $\text{match}(U, V) = \bot$.
*Otherwise,* $\text{match}(U, V)$ *is a feature structure containing features* $(A : Z)$ *such that* $A$ *is a feature name present in* $U$ *with the value* $X$*. If* $A$ *is present in* $V$ *with the value* $Y$*, then* $Z = \text{match}(X, Y)$ *else* $Z = X$.

**$U$ and $V$ are equal to the same atomic value $W$**
$\text{match}(U, V) = W$

**other cases:**
$\text{match}(U, V) = \bot$

For defining the result $unif(U, V)$ of the unification between two feature values $U$ and $V$, we take again Definition 3.2 by changing the second case as follows:

$$\begin{bmatrix} init-subj : \begin{bmatrix} cat : \top \end{bmatrix} \\ init-obj : \begin{bmatrix} exists : + \\ cat : \top \end{bmatrix} \\ passivable : + \\ mode : pastp \\ tense : nil \end{bmatrix}$$

Figure 4: Morpho-syntactic profile of the terminal module *agent-passive*

**$U$ and $V$ are both feature structures:**
*If $U$ and $V$ include the same feature $A$ with the values $X$ and $Y$ and* $\text{unif}(X,Y) = \bot$, *then* $\text{unif}(U,V) = \bot$.
*Otherwise,* $\text{unif}(U,V)$ *is a feature structure constituted of features $(A : Z)$ such that $A$ is a feature present in $U$ or $V$; if $A$ is present in both $U$ and $V$ with the values $X$ and $Y$, then $Z =$* $\text{unif}(X,Y)$ *else $Z$ is the value of $A$ in $U$ or $V$.*

The labelling of modules with morpho-syntactic profiles must be compatible with the inheritance relation in the following sense: if a module $D_1$ inherits from a module $D_2$, its profile $P_1$ must match the profile $P_2$ of $D_2$ according to Definition 3.2.
Figure 2 shows an example of morpho-syntactic profiles that decorate some syntactic modules of the grammar $G_{verb}$. Only the specific part of the profiles are represented with their modules with respect to the part which is inherited. The complete profile is then computed by unifying all profiles which label the super-classes with this specific part. When no label is associated with a module, the implied profile is the empty feature structure. For instance, according to these conventions, the morpho-syntactic profile of the terminal module *agent-passive* has the shape described by Figure 4 [3]. This profile means that the syntactic construction described by the module *agent-passive* only applies to transitive verbs with a passive form and an initial subject (impersonal verbs are discarded) and these verbs must be used at the past participle.
The mechanism for anchoring the modular grammar lexically will be complete if it allows the transfer of some features from the lexemes to the modules of the grammar. Usually, this mechanism resorts to a notion of anchor. For instance, TAG trees have a special node, their main anchor, at which all features of the lexeme will be assigned.
In our framework, we can design a more flexible mechanism for taking into account the fact that the morpho-syntactic properties of a lexeme can influence different nodes of the corresponding modules in the grammar at the same time. For instance, if the lexeme is a verb, the type of its subject, noun phrase or clause, will determine the feature *cat* of the node representing the initial subject (called *init-subj* in Figure 2) whereas its mode will concern the node representing the bare verb (called *vanc* in Figure 2). For this, we assume that the morphological and syntactic properties of any lexeme are described in the shape of a morpho-syntactic profile in the sense given by Definition 3.1. This profile is used for selecting the appropriate lexical modules from the grammar: a module will be selected by a lexeme if the profile of the lexeme matches the profile of the module according to Definition 3.2.
With this system, the transfer of features from the profiles of lexemes to the nodes of the descriptions present in the modules can be established by a mechanism of co-indexation between *external features* and *internal features*. For a given module, the external features compose its morpho-syntactic profile whereas its internal features are assigned to the nodes of the description that constitute its contents. For example, take the module *init-subj-real* of Figure 2. The external feature *cat* of the profile is indexed with the same integer 1 as the internal feature *cat* of the node *init-subj*. As soon as the profile of a lexeme matches the profile of the module *init-subj-real*, both external and internal features *cat* are instantiated by the same value. One can find other examples of internal features co-indexed with external features in Figure 2; they are drawn in bold type.
The addition of indices to features of morpho-syntactic profiles do not change the operations of matching and unification if we forbid re-entrancy.
To describe the process which generates and anchors the appropriate lexical modules for a given lexeme, let us take a concrete lexeme. For instance, the morpho-syntactic profile of the French past participle *mangé* can be represented in a simplified way by Figure 5. Among the 31 lexical modules of the grammar $G_{verb}$, this profile selects 24 of them and in particular the module presented in Figure 3. Matching the profile of the lexeme with the profile of a module entails the instantiation of the co-indexed external and internal features. For the lexical module of Figure 3, this concerns the features *phon*, *init-obj.cat*, *pers*, *num* and *gen* which are instantiated by the respective values *mangé*, *np*, $1 \vee 2 \vee 3$, *sg* and *m*.
Of course, the whole syntactic lexicon is not built by iterating this operation for all words of the language. Usually, syntactic lexicons are built by grouping words into families according to sub-categorization frames; then, syntactic structures are computed not for single words but for entire families of words. The use of feature structures allows one to envisage this way of building syntactic lexicons inside our frame-

[3] For the sake of uniformity in the presentation of feature structures, the feature *tense* always goes with the feature *mode* and when a mode has no tense, the feature *tense* takes the value *nil*.

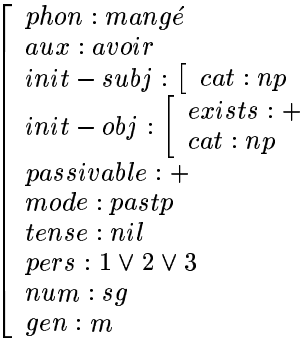


Figure 5: Morpho-syntactic profile of the past participle *mangé*

work in a very flexible manner. A family of words can be characterized by its morpho-syntactic profile in the same way as a word and then the process of selecting lexical modules from the grammar is performed in the same way.
Of course, from these general principles to a concrete organization of grammars and lexicons, it remains a lot of questions to solve but these questions go beyond the scope of our paper.

## Conclusion

In this paper, we have described a formal framework for the representation of a class of lexicalized grammars, the main advantage of which is a two-dimensional modularity. Horizontally, it allows one to split a grammar into modules structured by relations of inheritance, positive and negative co-occurrence in a uniform way. Vertically, we can distinguish three relatively independent levels in the grammar:

- the level of the architecture of the grammar which is given by the three kinds of relations between modules: inheritance, disjunctive inheritance and co-occurrence;
- the level of the contents of the modules in the shape of tree descriptions;
- the level of the morpho-syntactic profiles which decorate the modules of the grammar.

These three levels can be treated and modified in a relatively independent way. Since the framework applies to all formalisms which are expressible with tree descriptions, it would be interesting to measure to what extent these three levels are independent from the choice of the formalism. A way of answering this question consists in experimenting with the construction of wide coverage grammars for various formalisms.

## References

M.-H Candito. 1999. *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien.* Thèse de doctorat, Université Paris 7.

B. Carpenter. 1992. *The logic of typed feature structures.* Cambridge University Press, Cambridge.

J. Carroll, N. Nicolov, O. Shaumyan, M. Smets, and D. Weir. 1998. The LEXSYS Project. In *4th International Workshop on Tree Adjoining Grammars and Related Frameworks*, pages 29–33.

R. Evans and G. Gazdar. 1996. DATR: a Language for Lexical Knowledge Representation. *Computational Linguistics*, 22(2):167–247.

R. Evans, G. Gazdar, and D. Weir. 1995. Encoding Lexicalized Tree Adjoining Grammars with a nonmonotonic inheritance hierarchy. In *33rd Meeting of the Association for Computational Linguistics (ACL'95)*, pages 77–84.

A. Nasr. 1995. A formalism and a parser for Lexicalised Dependency Grammars. In *4th Int. Workshop on Parsing Technologies.* State University of NY Press.

G. Perrier. 2001. Intuitionistic proof nets as models of directed acyclic graph descriptions. In *8th International Conference LPAR 2001*, volume 2250 of *LNCS*, pages 233–248. Springer Verlag.

Carl J. Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar.* University of Chicago Press.

O. Rambow, K. Vijay-Shanker, and D. Weir. 1995. D-Tree Grammars. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 151–158.

O. Rambow, K. Vijay-Shanker, and D. Weir. 2001. D-tree Substitution Grammars. *Computational Linguistics*, 27(1):87–121.

J. Rogers and K. Vijay-Shanker. 1994. Obtaining Trees from their Descriptions: an Application to Tree Adjoining Grammars. *Computational Intelligence*, 10(4):401–421.

K. Vijay-Shanker. 1992. Using Description of Trees in a Tree Adjoining Grammar. *Computational Linguistics*, 18(4):481–517.

Research Group XTAG. 2001. A Lexicalized Tree Adjoining Grammar for English. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.