Metagrammars: a new implementation for FTAG

Sébastien Barrier, Nicolas Barrier
Laboratoire de linguistique formelle
Case Postale 7031
2, place Jussieu 75251 Paris Cedex 05, France
{sebastien,nicolas}.barrier@linguist.jussieu.fr

Abstract

This paper describes work on creating elementary trees for adjective and predicative noun families (Barrier, 2002; Barrier and Barrier, 2003) using Metagrammars, for the FTAG grammar (Abeillé, 1991; Abeillé, 2002). Based on the Candito's work on Metagrammars (Candito, 1996; Candito, 1999a), it adds a fourth dimension, specially designed for word order specification.

1 The metagrammar compiler

Metagrammars represent a TAG as a multiple inheritance network, whose classes specify syntactic properties. An important aspect of classes is that they are all related to one another. Inheritance enables classes that are logically related to one another to share the behaviors and attributes that they have in common.

Our metagrammar imposes an overall organization for syntactic data and formelizes the well-formedness conditions on elementary tree sketches (Vijay-Shanker and Schabes, 1992; Rogers and Vijay-Shanker, 1994).

Each syntactic property of the hand-written inheritance network – the hierarchy – is declared as a complete syntactic set of partial descriptions. Those partial descriptions can be seen as syntactic constraints (dominance, linear precedence, ...) which may leave underspecified the relation between two nodes – the relation can be further explained by adding constraints in sub-classes of the network.

In concrete terms, data are defined as global variables augmented with specific meta-features, constraining for instance the possible part of speech of a node, or function for argument ones.

Structures sharing the same initial subcategorization frame may only differ in the surface realization of the final syntactic function of the arguments nodes, according to their redistribution.

The hand-written hierarchy was initially divided into 3 dimensions, and has been more recently extended to 4 dimensions (Barrier and Barrier, 2003):

- Dimension 1 : initial subcategorization.
- Dimension 2 : redistribution of functions.
- Dimension 3 : Surface realizations of syntactic functions.
- Dimension 4 : word order specification of surface realizations of syntactic functions.

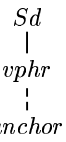
Contrary to (Vijay-Shanker and Schabes, 1992), we do not have explicit lexical rules: diathesis alternations are represented by classes of dimension 2, whereas marked and unmarked cases are represented by classes of dimension 3. Dimension 4 allows to express word order in a directly readable and not confusing way: classes of dimension 1 and 2 were clearly inappropriate (word order has nothing to deal with declaration of grammatical functions), whereas classes of dimension 3 couldn't predict the existence or the lack of another argument.

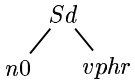
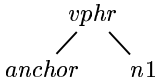
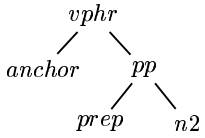
In order to automatically generate elementary trees, the compiler creates additional classes, named "crossing-classes". Each crossing class inherits from one class of dimension 1, then inherits from one class of dimension 2, and lastely inherits from classes of dimension 3, representing the realizations of every function of the final subcategorization. Classes of dimension 4 are not crossed automatically: all the crossings are declared manually by the metagrammar's writer so that he can only express the crossings, which are necessary. Crossings are accordingly only done when all the relevant classes are involved.

Finally each crossing class is translated into one or more elementary trees, satisfying all inherited constraints.

TAG+7: Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms.
May 20-22, 2004, Vancouver, BC, CA.
Pages 132-139.

Dimension 1		
The class (DI-TRANS) inherits from (SUBJ), (OBJ) and (IND-OBJ)		
(SUBJ) Class	(OBJ) Class	(IND-OBJ) Class
Variable $arg0$ stands for $NP_0 \downarrow$ and bears Subject function	Variable $arg1$ stands for $NP_1 \downarrow$ and bears Object function	Variable $arg2$ stands for PP_2 and bears Indirect Object function

Dimension 2
The class (NO-REDIS) inherits from (VB-MORPH)
(VB-MORPH) Class
Variable Sd stands for S Variable $vphr$ stands for VP Variable $anchor$ stands for $V\diamond$ 

Dimension 3		
The class (SUBJ-CAN) inherits from (POS-SUBJ) (POS-SUBJ) allows to group all the realizations of the Subject	The class (OBJ-CAN) inherits from (POS-OBJ) (POS-OBJ) allows to group all the realizations of the Object	The class (IND-OBJ-CAN) inherits from (POS-IO) (POS-IO) allows to group all the realizations of the Indirect Object
(SUBJ-CAN) Class	(OBJ-CAN) Class	(IND-OBJ-CAN) Class
Variable $n0$ bears Subject function 	Variable $n1$ bears Object function 	Variable pp bears Indirect Object function Variable $Prep$ stands for $to\diamond$ Variable $n2$ stands for $NP_2 \downarrow$ 

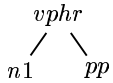
Dimension 4
(OBJ<IO) Class
This class will be used when both (OBJ-CAN) and (IND-OBJ-CAN) will appear


Table 1: Verbal hierarchy for di-transitive verbs

An inheritance hierarchy such as the one shown in Table 1, allows to represent the relevant tree sketch for the english sentence *Max gives a book to Peter*. It will be compiled out of an initial subcategorization with subject, direct object and indirect object (dimension 1), an active canonical redistribution (dimension2), canonical realizations of subject, direct object and indirect object (dimension 3), and a special word order, specifying indirect object follows direct object (dimension 4).

The compiler will automatically cross (DI-TRANS), (NO-REDIS), (SUBJ-CAN), (OBJ-CAN) and (IND-OBJ-CAN) classes. As (OBJ-CAN) and (IND-OBJ-CAN) are crossed, (OBJ<IO) will also be crossed with the other classes. The resulting tree sketch will be the conjunction of all quasi-tree descriptions contained in each class. The nodes with same variables will unify; the variables with same function will also unfify.

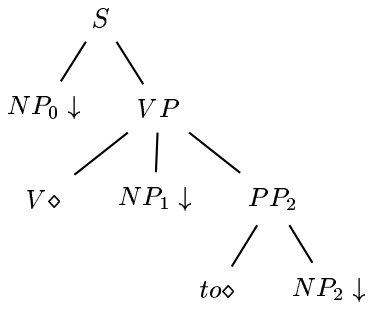


Figure 1: Elementary tree for *Mary gives a book to Peter*

Note that the metagrammar compiler makes use of variables as global variables. There is no way to use local variables. Linear precedence can't be expressed without reference to dominance.

The Metagrammar compiler we use was first developed by (Candito, 1999a) in Lucid Common Lisp and has been in part reimplemented in CLISP by (Barrier, 2002). It generates tree sketches in both XTAG or TAGML2 format with t-feature structures (see below).

2 Choices and implementation

2.1 Linguistics principles and general choices

As mentionned in (Abeillé et al., 2000), FTAG elementary trees respect the following well-formedness principles :

- Strict lexicalization: all elementary trees are anchored by at least one lexical element (the empty string cannot anchor a tree by itself)
- Semantic consistency: no elementary tree is semantically void
- Semantic minimality: elementary trees correspond to no more than one semantic unit
- Predicate argument cooccurence principle : an elementary tree is the minimal syntactic structure that includes a leaf node for each realized semantic argument of the anchor(s)

Semantic minimality and consistency imply that function words appear as co-anchors.

Most of the linguistic analyses follow those of (Abeillé, 1991; Abeillé, 2002) (except that clitic arguments are substituted and not adjoined), complemented by (Candito, 1999a). We dispense with most empty categories, especially in the case of extraction. Semantically void (or non autonomous) elements, such as complementizers, argument marking prepositions or idiom chunks are co-anchors in the elementary tree of their governing predicate.

Passive is characterized by a particular morphology, with a substitution node for the auxiliary verb. Causative constructions are analyzed as complex predicates, with a flat structure, with a substitution node for the causative verb.

For oblique complements, we distinguish between a-objects, de-objects, locatives and other prep-objects, depending on the pronominal realization of the complement.

2.2 New families for FTAG

We have chosen not to reuse Candito's verbal hierarchy because of inconsistencies: it was not fully documented and hard to understand. Some classes of dimension 3 inherit from classes of dimension 1 or 2, which is normally not allowed by the metagrammar concept. Furthermore, this verbal hierarchy contains some empty classes.

We developed 34 new families: 16 adjectival families allow us to create 2690 tree sketches, whereas 18 support verb families allow us to create over 10.000 tree sketches.

2.2.1 Adjectival families

We regard the adjective as the local head of the adjectival predicate, and consider object predicate's constructions as an alternative of causative constructions. An unique family provides tree sketches for both predicative and attibutive adjectives, so that we can encode relative clauses or clitics for different kind of adjective complements. We describe the concept of subject as the category modified by the adjective. No object function can be found: all the complements of the adjectival predicate are always indirect ones.

Our grammar covers the following types of redistribution :

- Predicative adjective : Jean est barbu

- Causative : Sarah Vaughan rend les gens heureux
- Passive causative : Des gens sont rendus heureux
- Impersonal causative passive : Il est rendu impossible de faire cela
- Impersonal : Il est inacceptable de dormir ici
- Attributive adjective : Un homme heureux

The syntactic realizations covered are canonical position, extraction (cleft and relativized), clitic or non-realized.

2.2.2 Predicative noun families

The lexical head is only the predicative noun, whereas the support verb is substituted into the tree associated with the noun. This differs from the light verb families from XTAG (and also from the previous versions of FTAG) where the verb and the noun both anchor the tree. An unique family provides tree sketches for support verb constructions and nominal phrases.

Our grammar covers the following types of redistribution :

- Active: Max commet un crime contre Luc
- Passive: Un crime est commis par Max contre Luc
- Middle: Un crime se commet contre Luc en 5 minutes
- Causative: Léa fait commettre une crime à Max contre Luc
- Passive Impersonal: Il est commis un crime par Max contre Luc
- Impersonal Middle: il se commet un crime toutes les 5 minutes
- Nominal phrase: le crime de Max contre Luc

The syntactic realizations covered are canonical position, extraction (cleft, relativized and questionned), clitic and non-realized.

Datasheet for adjective and predicative noun hierarchies can be found at the end of this article. Each page represents Dimension 1, 2 and 3. Dimension 4 is not shown since it is not particular to these hierarchies. It is specially used for clitic word order.

2.3 Main difficulties

A typical error consists in encoding more than a class expects. One may de facto limit the syntactic properties sharing. Metagrammars do not exempt from studying syntactic phenomena but force ones to understand what classes share with in terms of syntactic properties. Since arguments are realized as independent functions the metagrammar's writer not only has to find a way to arrange them correctly inside the tree but has to encode his classes so that they can be reused for another category.

Another place metagrammars and inheritance networks go wild is in making very deep hierarchy. It can be very tedious to look many levels up to the tree to find out what a particular inherited variable is supposed to be: it is easy to create complex hierarchy that is hard to understand, even for the metagrammer's writer who created it. Inheritance, just like many other elements of OOP is just a tool. If the problem calls for it, it seems interesting to use it, but one doesn't see it as a solution to all problems. With proper usage, metagrammars will save the writer from retyping and will show him that different linguistic objects are related.

3 Current and future work

To take advantage of the hierarchical representation of tree sketches within our metagrammar, we characterize tree sketches as feature structures we call t-feature structures (Abeillé et al., 1999).

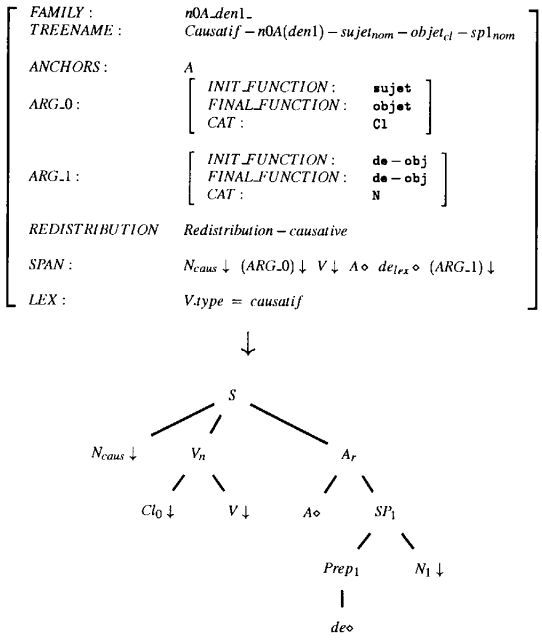
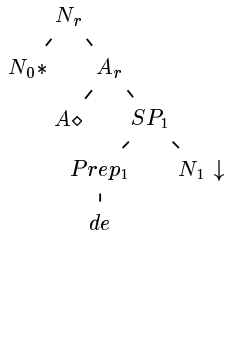
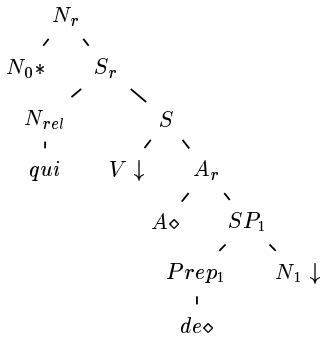


Figure 2: Tree sketch for a causative construction used for an adjectival predicate

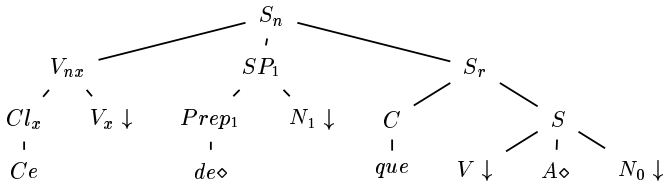
While the automatic generation of the grammar insures consistency, errors may still propagate but on a larger scale, with dramatic effects if it remains undetected. These feature-structures keep track of the successive mapping steps that are performed during the genera-



Un homme fier de sa fille
A man proud of his daughter

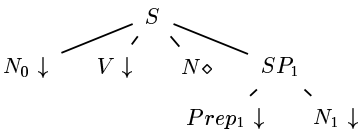


Max qui est fier de sa fille
Max who is proud of his daughter

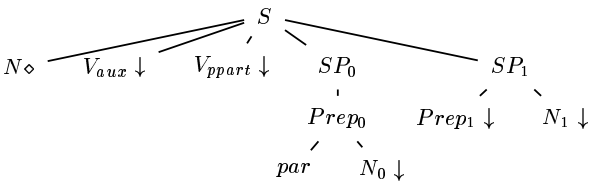


C'est de sa fille qu'est fier Max
It is of his daughter that Max is proud

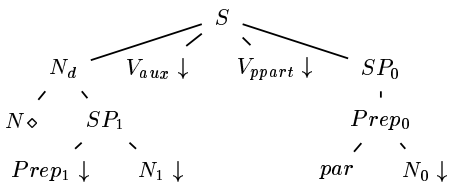
Table 2: Some elementary trees taken from *n0A(den1)* family



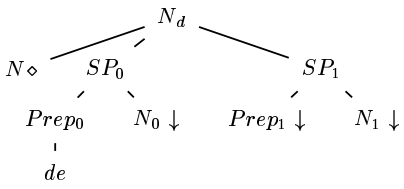
Max commet un crime contre Luc
Max commits a crime against Luc



Un crime est commis par Max contre Luc
A crime is commited by Max against Luc



Un crime contre Luc est commis par Max
A crime against Luc is committed by Max



Le crime de Max contre Luc
Max's crime against Luc

Table 3: Some elementary trees taken from the $n0vN(pn1)$ family

tion process.

Characterizing tree sketches as a combination of features allows us to refer to a set of tree sketches simply by under specifying a feature structure.

It could also be interesting to merge all the hierachies into one. But this will probably be a hard task[1]. Each Metagrammar's writer has indeed his own view of specific problems.

We hope to evaluate our grammar in few weeks by using treebank 'Le Monde' developed at Paris 7 University (Abeillé et al., 2003).

References

Anne Abeillé and Owen Rambow. 2000. *Tree Adjoining grammars*. CSLI Publications.

Anne Abeillé, Marie-Hélène Candito, and Alexandra Kinyon. 1999. Ftag: current status and parsing scheme. In *Vextal'99*.

Anne Abeillé, Marie-Hélène Candito, and Alexandra Kinyon. 2000. The current status of FTAG. In *Proceedings of TAG+5*.

Anne Abeillé, Nicolas Barrier, and Sébastien Barrier. 2001. FTAG, une grammaire LTAG du français. Technical Report 1.0.

Anne Abeillé. 1991. *Une grammaire lexicalisée d'arbres adjoints pour le français*. Ph.D. thesis, Université Paris 7.

Anne Abeillé. 2002. *Une grammaire électronique du français*. CNRS Editions.

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In A. Abeillé, editor, *Treebanks: building and using parsed corpora*, pages 165–188. Kluwer academic publishers.

Sébastien Barrier and Nicolas Barrier. 2003. Une métagrammaire pour les noms prédicatifs du français. In *TALN 2003*.

Nicolas Barrier. 2002. Une métagrammaire pour les adjectifs du français. In *TALN 2002*.

Davy Boonen. 2001. Le prédicat adjectival en ftag. Master's thesis, Université Paris 7.

Marie-Hélène Candito. 1996. A principle-based hierarchical representation of ltag. In *Proceedings* 15^{th} *COLING*.

Marie-Hélène Candito. 1999a. *Représentation modulaire et paramétrable de grammaires électroniques lexicalisées, application au français et à l'italien*. Ph.D. thesis, Université Paris 7.

[1]Of course, it does not mean all the new tree sketches cannot be combined into one grammar.

Marie-Hélène Candito. 1999b. Un outil multilingue de construction semi-automatique de grammaire d'arbres adjoints. In *TAL*, volume 40.

Laurence Danlos. 1992. Support verb constructions. In *Journal of French Linguistic Study*.

Laurence Danlos. 1998. GTAG, un formalisme lexicalisé pour la génération inspiré de TAG. In *TAL*, volume 39.2.

Bertrand Gaiffe, Benoît Crabbé, and Azim Roussanaly. 2002. A new metagrammar compiler. In *Proceedings of TAG+6*.

Kim Gerdes. 2002. *Topologie et grammaires formelles de l'allemand*. Ph.D. thesis, Université Paris 7.

Jacqueline Giry-Schneider. 1978. *Les nominalisations en français*. Droz. Genève-Paris.

Jacqueline Giry-Schneider. 1987. *Les prédicats nominaux en français*. Droz. Genève-Paris.

Jan Goes. 1999. *L'adjectif entre nom et verbe*. Duculot.

Maurice Gross. 1981. Les bases empiriques de la notion de prédicat sémantique. In *Langages*, volume 63.

Zellig Harris. 1968. *Mathematical structures of language*. Wiley-Interscience.

Alexandra Kinyon. 2000. Even better than supertags: introducing hypertags. In *Proceedings of TAG+5*.

Michèle Noailly. 1990. *Le substantif épithète*. PUF. Paris.

Michèle Noailly. 1999. *L'adjectif en français*. OPHRYS. Paris.

J. Rogers and K. Vijay-Shanker. 1994. Obtaining trees from their descriptions, an application to tree adjoining grammar. In *Computational intelligence*, volume 10.4.

K. Vijay-Shanker and Y. Schabes. 1992. Structure sharing in lexicalized tree adjoining grammar. In *Proceedings of COLING-92*.

Annexe A - Datasheet for Adjectives

Family	Example	Family	Example
n0A	*Jean est barbu* John is bearded	n0A(as1)	*Jean est attentif à ne blesser personne* John is cautious not to hurt anyone
n0A(pn1)	*Jean est fort en histoire* John is good at history	n0A(des1)	*Jean est certain qu'ils viendront* John is convinced they will come
n0A(an1)	*Jean est sourd à cette proposition* John is deaf to this proposal	n0A(an1)(des2)	*Jean est reconnaissant à Marie de faire ses devoirs* John thanks Mary for doing his homework
n0A(den1)	*Jean est amoureux de Marie* John is in love with Mary	s0A	*Prendre le thé sur la pelouse est inacceptable* Having tea out on the lawn is unacceptable
n0A(an1)(pn2)	*Jean est supérieur à Marie en histoire* John is higher than Mary at history	s0A(pn1)	*Prendre le thé est bon pour la santé* Having tea is good for health
n01(an1)(den2)	*Jean est redevable de 10€ à Marie* John owes Mary 10€	s0A(ps1)	*Faire du sport est bon pour éviter les crises cardiaques* Doing sport is good to prevent heart attacks
n0A(den1)(pn2)	*Jean est quitte de ses dettes envers la société* John has paid his debt to society	s0A(an1)	*Prendre le thé est nécessaire aux hommes* Having tea is necessary to men
n0A(ps1)	*Boire du thé est bon pour le mal de tête* Having tea is good for headaches	s0A(den1)	*Faire du sport est indépendant de vos autres activités* Doing sport is independant from your other activities

Table 4: Adjectival families

	Initial subject				
Construction	N	Cl	S	Redistribution	Example
Predicative adjective	+	+	+	No redistribution	Jean est barbu
Causative	+	+	-	Subject > Object Causer > Subject	Sarah Vaughan rend les gens heureux
Passive causative	+	+	+	Causer > Par_obj Object > Subject	Des gens sont rendus heureux (par Sarah)
Impersonal causative passive	+	+	+	Causer > empty Impersonnal > Subject	Il est rendu impossible de faire cela
Attributive adjective	+	-	-	Subject > Subject_epi	Un homme heureux
Impersonal	-	-	+	Subject > Sentencial indirect cmpl Impersonal > Subject	Il est inacceptable de commettre des erreurs

Table 5: Redistribution frame for adjectives

	Surface realizations					
	Nominal	Clitic	Cleft	Sentencial	Relativized	Non-realized
Subject	Canonical Inverted	X	Nominal Sentencial	X	qui	
Prep-obj	X		Nominal Sentencial	X	X	X
A-obj	X	X	Nominal Sentencial	X	X	X
De-obj	X	X	Nominal Sentencial	X	dont	X
Prep-obj2	X		Nominal		X	X
De-obj2		X	Nominal	X	dont	X
Indirect Sentencial cmpl				X		
Predicative object	Anteposed Postposed	X				
Par-Obj	X					X

Table 6: Surface realization of syntactic functions for adjectives

Annexe B - Datasheet for Predicative Nouns

Family	Example	Family	Example
n0vN	*Max prend un bain* Max takes a bath	n0vPN(as1)	*Max a de la peine a dormir* Max has difficulty in sleeping
n0vN(an1)	*Max fait du chantage à Luc* Max blackmails Luc	s0vN	*Prendre le thé sur la pelouse fait scandale* Having tea out on the lawn scandalized people
n0vN(den1)	*Max fait la censure de cette page* Max censors this page	s0vN(den1)	*Prendre le thé sur la pelouse fait la joie de Luc* Having tea out on the lawn gives great pleasure to Luc
n0vN(loc1)	*Max fait un pélerinage à Lourdes* Max goes on a pilgrimage to Lourdes	s0vPN(den1)	*Faire du sport est à l'avantage de Max* Doing sport gives an advantage to Max
n0vN(pn1)	*Max commet un crime contre Luc* Max commits a crime against Luc	n0vN(den1)(an2)	*Max fait le récit de son histoire à Luc* Max gives an account of his story to Luc
n0vN(des1)	*Max a l'espoir de réussir* Max hopes he will succeed	n0vN(den1)(pn2)	*Max fait la division de 4 par 2* Max divides 4 by 2
n0vN(ps1)	*Max fait un effort pour rester calme* Max makes an effort to stay calm	n0vN(den1)(loc2)	*Mac fait une expédition de livres en Somalie* Max send books in Somalia
n0vPN(pn1)	*Max est en colère contre Luc* Max is angry with Lux	n0vN(pn1)(pn2)	*Max fait une plaisanterie sur Luc avec Léa* Max makes a joke with Léa on Luc
n0vPN(den1)	*Max est dans l'ignorance de cet incident* Max is unaware of this event	n0vN(an1)(des2)	*Max a donné l'ordre à Luc de partir* Max has ordered Luc to go

Table 7: Predicative nouns families

Construction	Redistribution	Example
Passive	object > subject subject > par_object	Un crime est commis par Max contre Luc Un crime contre Luc est commis par Max
Middle	subject > empty object > subject	Un crime se commet contre Luc en 5 minutes Un crime contre Luc se commet en 5 minutes
Causative-A	subject > empty causer > subject	Léa fait commettre un crime à Max contre Luc
Impersonal Middle	subject > empty Impers > subject	Il se commet un crime toutes les 5 minutes
Impersonal Passive	subject > par_object impers > subject	Il est commis un crime par Max contre Luc Il est commis un crime contre Luc par Max
Nominal phrase	object > empty prep_object > cdn	Le crime de Max contre Luc

Table 8: Redistribution frame for predicative nouns

	Surface realizations						
	Nominal	*Clitic*	*Cleft*	*Sentencial*	*Relativized*	*Questionned*	*Non-realized*
Subject	Canonical Inverted	X	Nominal	X	qui	X	
Predicative Noun	X		Nominal		que		
Prep Obj	X		Nominal	X	X	X	X
A-Obj	X	X	Nominal		X	X	X
De-obj	X	X	Nominal		dont	X	X
Prep-Obj2	X		Nominal		X	X	X
A-Obj2	X	X	Nominal		X	X	X
Indirect sentencial cmpl				X			
Par-Obj	X		Nominal			X	X

Table 9: Surface realization of syntactic functions for predicative nouns