FRIGRAM: a French Interaction Grammar

Guy Perrier[1] *and Bruno Guillaume*[2]
[1]LORIA, Université de Lorraine, Nancy, France
[2]LORIA, Inria Nancy Grand-Est

ABSTRACT

We present FRIGRAM, a French grammar with a large coverage, written in the formalism of Interaction Grammars. The originality of the formalism lies in its system of polarities which expresses the resource sensitivity of natural languages and which is used to guide syntactic composition. We present the principles underlying grammar design, highlight its modular architecture and show that the lexicon used is independent of the grammar formalism. We also introduce the "companion property", and show that it helps to enforce grammar consistency.

Keywords: formal grammar, model theoretic syntax, polarity, Interaction Grammar.

1 INTRODUCTION

Following the seminal work initiated on Categorial Grammars (CG) by Lambek (1958), many other grammatical formalisms were proposed to describe the syntax of natural language. Apart from CG, the most well-known ones are TAG (Joshi *et al.* 1975), LFG (Bresnan 2001) and HPSG (Pollard and Sag 1994). These formalisms have several advantages. First, they allow for a detailed encoding of linguistic knowledge. Second, they can be used to investigate formal properties of natural language or to study linguistic hypotheses by testing them on real linguistic utterances. Third, grammars written using these formalisms can be used in more complete systems where syntax modeling is required either within a parsing or a generation application.

All these formalisms use a finite set of elementary structures and some mechanisms for composing these elementary structures to produce syntactic structures for larger utterances of a natural language. A large coverage system based on these approaches necessarily requires a huge number of elementary structures hence the development and the maintenance of such systems is a challenge and requires a lot of manual work. Among existing works to develop large coverage systems, mainly for English but also for some other languages, we can cite the ParGram (Butt *et al.* 2002) project (for LFG), the DELPHIN (Oepen *et al.* 2002) project (for HPSG) and XTAG (XTAG Research Group 2001) (for TAG).

We aim to conduct similar work within the framework of Interaction Grammars (IG), a formalism first introduced in Perrier (2000) and presented in more detail in Guillaume and Perrier (2009). IG combines a flexible view of grammars as constraint systems with the use of a polarity system to control syntactic composition. This polarity system expresses the saturation state of partial syntactic structures and their ability to combine.

The present paper reports on the construction of a syntactic resource for the French language and shows that it is possible to build a wide coverage IG grammar of French which encodes fine-grained linguistic knowledge.

In this grammar (called FRIGRAM), the syntax of French sentences is described by dependency structures which contain the usual surface syntactic dependencies but also by additional dependency relations which contain the information needed to produce a deeper syntactic analysis. These additional relations are: infinitival subjects, participial subjects and pronoun antecedents that are syntactically predictable.

The main challenge is to guarantee and maintain the consistency of the grammar while aiming for large coverage. To this end, we resort to the following means:

- a modular organization of the grammar in a hierarchy of classes which captures linguistic generalizations,
- well-formedness principles imposed on the elementary structures of the grammar,
- a strict separation between grammar and lexicon with a lexicon that is independent of the particular grammatical formalism used,

- the use of the *companion property* to help checking grammar consistency.

The paper is structured as follows. We start with a brief presentation of IG. We then go on to explain the different points just mentioned. We conclude with a comparison with other French grammars and discussion about the evaluation of the grammar.

2 INTERACTION GRAMMARS

IG is a grammatical formalism which describes the syntax of natural language using two notions: *tree description* and *polarity*. For a complete presentation of the formalism, the reader is referred to Guillaume and Perrier (2009).

2.1 *Parse trees*

In IG, the syntax of sentences is modeled using constituency-based parse trees. An example of such a parse tree for the sentence *montrez-le !* 'show it!' is given in Figure 1.[1] Parse trees are totally ordered trees. Each node represents a constituent and carries a feature structure which encodes morpho-syntactic properties. All features in parse trees are written with the ':' symbol like in `[f:v]` in contrast with features in tree description that are written with other symbols (see below). Leaves of parse trees can be either anchors carrying lexical units (nodes ⟨1.2.1⟩, ⟨1.2.2.1⟩, and ⟨2.1⟩) or empty nodes (⟨1.1⟩ and ⟨1.3⟩).

Empty nodes represent constituents that are not directly realized in the syntax. In FRIGRAM, empty nodes are used in several ways. The empty node ⟨1.1⟩ is an example of an argument which is not syntactically realized (feature `[empty_type:arg]`). This is the case for infinitival and imperative subjects, as well as for some infinitival objects (tough movement). In Figure 1, the node ⟨1.1⟩ represents the non-expressed subject of the imperative verb *montrez*. The empty node ⟨1.3⟩ is an example of a trace of an argument which is moved from its canonical position, this trace is marked (feature `[empty_type:track]`). In Figure 1, the node ⟨1.3⟩ is the trace of

[1] We suppose that the hyphen in *montrez-le !* is removed at the tokenization stage.

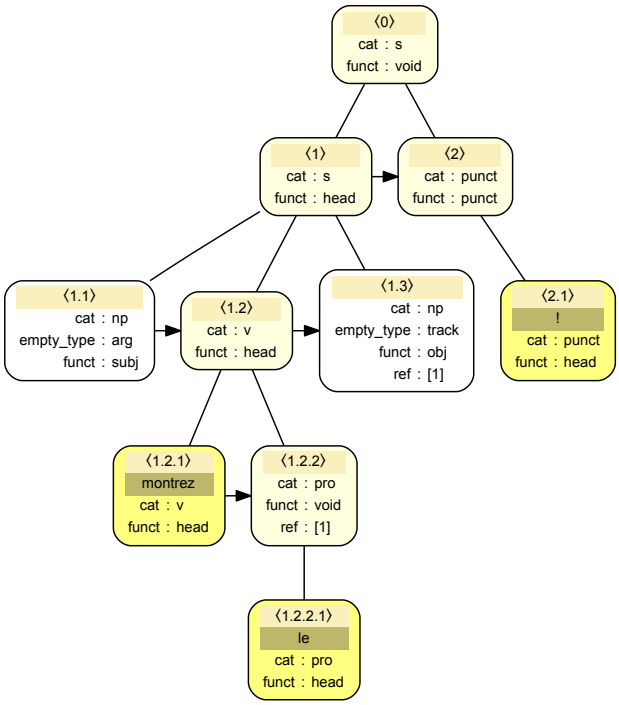


Figure 1: Parse tree representing the syntax of the sentence *montrez-le !*

the object clitic pronoun *le*;[2] the link between the pronoun and the extracted position is encoded by feature sharing: both constituents contain the feature `[ref:[1]]`. This mechanism is used in all cases of extraction, subject inversion, and cliticization of arguments.

2.2 *Tree descriptions*

To produce the parse trees described above, IG relies on a model theoretic syntax approach of natural languages (Pullum and Scholz 2001). Parse trees are defined as models of a more general notion of tree description (introduced by Rogers and Vijay-Shanker 1994). In this view, the basic objects of the grammar are not trees but properties that are used to describe them, in other words, *tree descriptions*. This approach is flexible and allows expressing elementary properties in independent ways and combining them freely. A tree description can be viewed ei-

[2] For consistency, all clitics are treated as moved arguments even if, as in the short sentence *montrez-le !*, the clitic *le* seems to be in a canonical place.

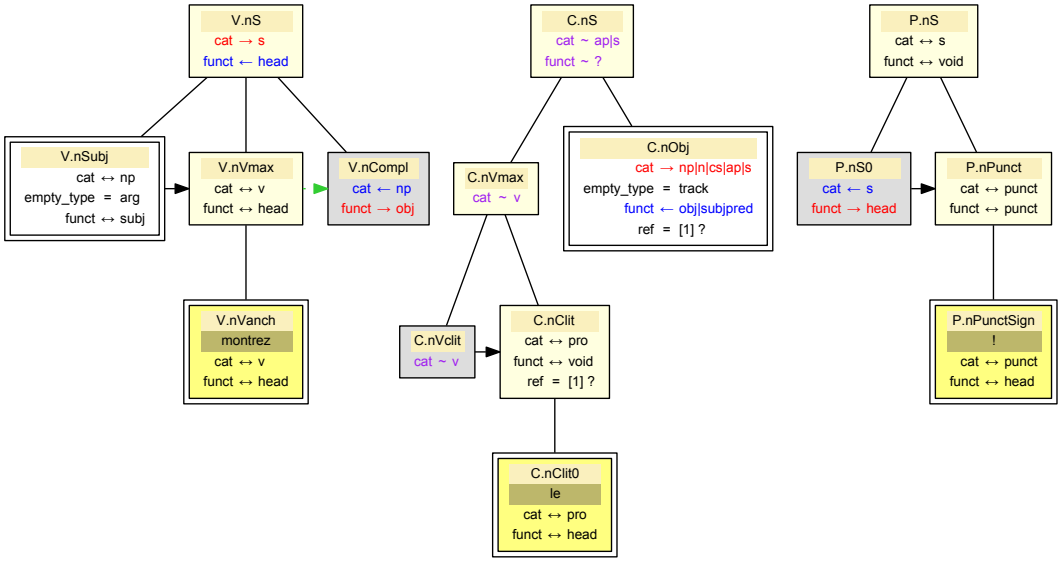


Figure 2: Polarized tree description associated with the sentence *montrez-le !* by the grammar FRIGRAM.

ther as an underspecified tree, or as the specification of a tree family, each tree being a model of this specification.

Figure 2 gives an example of a tree description[3] which is associated with the sentence *montrez-le !*. This description is composed of three connected components associated with the three lexical units *montrez*, *le*, and *!* occurring in that sentence.

Formally, a tree description is a finite set of nodes structured by two kinds of relations: *dominance* and *precedence*. Dominance relations can be immediate or underspecified. In the example, there are only immediate dominance relations represented with solid lines. Precedence relations can also be immediate or underspecified. They are represented with arrows; these arrows are solid and black (for immediate precedence) or dashed and green (for underspecified precedence).

Nodes are labeled with features describing their morpho-syntactic properties whereby a feature value is either an atom (like in a parse tree) or a disjunction of atoms. When a feature value is the disjunction of all elements of a domain, this value is denoted with "?".

[3] Note that the figures are simplified and only display a part of the full feature structures.

A co-indexation mechanism between feature values is also available at the description tree level (a common index $[n]$ is put before their values). For instance, the `ref` feature of node *C.nObj* shares its value with the `ref` feature of node *C.nClit* meaning that both constituents refer to the same semantic entity.

2.3 *Polarities*

Polarities are used in tree descriptions to describe the saturation state of incomplete syntactic trees. The set of features is partitioned into two subsets: the set of *resource sensitive* features and the set of *neutral* features. Neutral features are written as `[f=v]`. For instance, agreement properties are expressed with neutral features.

Polarities are attached to resource sensitive features that label description nodes. Given a feature name `f` and a feature value `v` (which may be either an atomic value or a disjunction of atomic values), the four kinds of polarized features and their meanings are:

- a *positive* feature `[f→v]` expresses an available resource which must be consumed;
- a *negative* feature `[f←v]` expresses an expected resource which must be provided; it is the dual of a positive feature; one negative feature must match exactly one corresponding positive feature to be saturated and conversely;
- a *saturated* feature `[f↔v]` expresses a linguistic property that needs no combination to be saturated;
- a *virtual* feature `[f~v]` expresses a linguistic property that needs to be realized by combining with either a saturated feature or a pair of a negative and positive features.

In Figure 2, node *V.nCompl* carries a negative `[cat←np]` and positive feature `[funct→obj]`, which represents the expected object noun phrase for the transitive verb *montrez*.

The virtual features in the three nodes *C.nVclit*, *C.nVmax*, and *C.nS* of the second connected component in Figure 2 represent the syntactic context required by the clitic pronoun *le*, namely, a verb *C.nVclit* occurring immediately before the pronoun to build the node *C.nVmax* with it.

The descriptions labeled with polarized features are called *polarized tree descriptions (PTDs)* in the rest of the article.

2.4 Grammars as constraint systems

An interaction grammar is defined by a finite set of PTDs, named Elementary PTDs (EPTDs) and generates a tree language. A parse tree (as defined in Section 2.1) belongs to the language if it is a model of a finite list of EPTDs in the sense given in Guillaume and Perrier (2009). Each node of the list of EPTDs is mapped to a node of the tree model through an interpretation function. When two nodes of EPTDs have the same image by the interpretation function, we say that they are *merged*. We also say that the features labeling these nodes are *merged*. Models can be saturated and/or minimal:

- A tree model is *saturated* if every positive feature $[\mathtt{f} \rightarrow \mathtt{v}]$ is merged with a dual feature $[\mathtt{f} \leftarrow \mathtt{v}]$ (and vice versa) and if every virtual feature is merged with either a saturated feature or a pair of a positive and a negative feature. Merging a positive (resp. negative) polarity with a saturated polarity or with another positive (resp. negative) polarity is forbidden. There is no saturation constraint on neutral features.
- A tree model is *minimal* if a minimum of information is added to the input EPTDs (no node, immediate dominance relation or feature that does not exist in the initial descriptions is added).

Parsing a sentence with a grammar G consists first of all in selecting an appropriate list of EPTDs from G. This selection step is easier if G is lexicalized. In that case, each EPTD has an anchor associated with a lexical unit of the language. This strongly reduces the search space for the EPTDs.

Then, the parsing process itself reduces to the resolution of a constraint system. It consists in building all models of the selected lists of EPTDs that respect the linear order of the words in the input sentence.

Figure 2 represented one of the possible selections of EPTDs from FRIGRAM for the sentence *montrez-le !*. The selection includes three EPTDs[4] which are considered as one single PTD with three connected components. Figure 3 shows the unique minimal and saturated model of the PTD. The parse tree of Figure 3 is the same as the one of Figure 1

[4] The three EPTDs are identified by a short name: in the order of their appearance in the sentence, *V* for the verb, *C* for the clitic pronoun, *P* for the punctuation. Nodes are then named with a prefixed notation like *C.nS*.

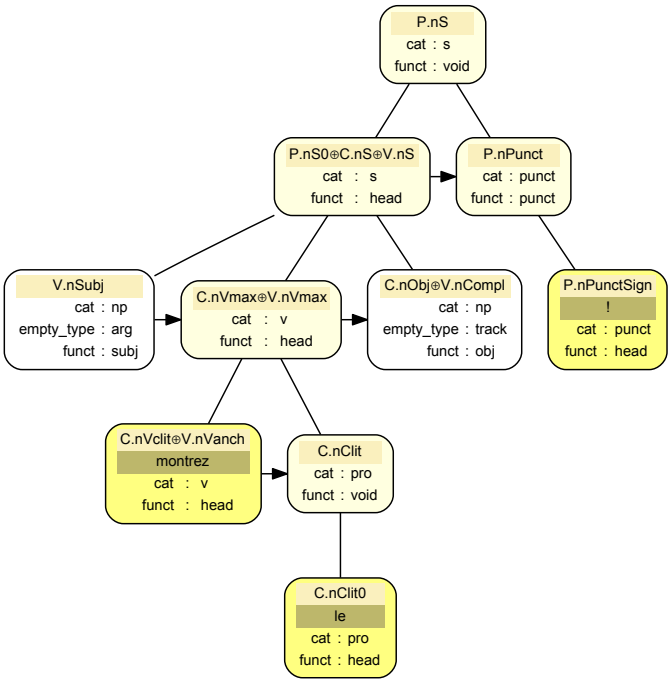


Figure 3: Tree model of the PTD shown in Figure 2 representing the syntax of the sentence *montrez-le !*

but with information about the interpretation function used to build the model: the set of nodes of the PTD of Figure 2 that are interpreted in the given node of the model is given in the first line of each node. For instance, the anchor *montrez* is the interpretation of the two nodes *C.nVclit* and *V.nVanch* of the PTD of Figure 2.

It can be checked that all features in the model are saturated. For instance, the feature `cat` of the node *P.nS0/C.nS/V.nS* is saturated because the feature `[cat → s]` of node *V.nS* is merged with the negative feature `[cat ← s]` of node *P.nS0* and the virtual feature `[cat ∼ ap|s]` of node *G.nS* is merged with the feature `[cat → s]` of node *V.nS*.

Other kinds on constraints are possible in PTDs. A node can be declared as *empty* (graphically represented with a white background, like nodes *V.nSubj* and *C.nObj* in Figure 2). It is then required that the image has an empty phonological projection, i.e., all leaves of the corresponding subtree are empty nodes. A node can be declared as *full* (graphically represented with a yellow background, like nodes *V.nS* and *C.nS* in Figure 2); it is then required that the image has a nonempty phonological projection: there is at least one anchor node

in the corresponding subtree in the model. Finally, a node can be declared to be *closed* (represented with a double rectangle, like nodes *C.nObj* and *P.PunctSign*) meaning that the set of its daughters is fixed. In the model, the node cannot have daughters that are not already present in the PTD.

2.5 *Parsing as a process of node merging controlled by polarities*

In an operational view of parsing, the building of a saturated and minimal model is performed step by step by refining the initial PTD with a merging operation between nodes, guided by one of the following constraints:

- neutralize a positive feature with a negative feature having the same name and carrying a value unifiable with the value of the positive feature;
- realize a virtual feature by combining it with a positive or saturated feature having the same name and carrying a value unifiable with the value of that virtual feature.

The constraints of the description interact with node merging to entail a partial superposition of their contexts represented by the tree fragments in which they occur. Let us illustrate this phenomenon with the parsing of the sentence *montrez-le !*, starting from PTD of Figure 2.

First, we try to saturate the virtual feature `[cat ∼ v]` of the *C.nVclit* node by merging it with the *V.nVanch* node. The final target model is a tree and thus, every node has one mother node at most. As a consequence, in the PTD, merging propagates to the ancestors of the two nodes *C.nVclit* and *V.nVanch*: nodes *C.nVmax* and *V.nVmax* and again propagates to nodes *C.nS* and *V.nS*. Figure 4 shows the resulting PTD.

In a second step, nodes *V.nCompl* and *C.nObj* are merged to saturate their polarities `[cat ← np]` and `[cat → np|n|cs|ap|s]` respectively. Figure 5 shows the PTD resulting from this second merging.

In a last step, the root of the left tree and the *P.nS0* node of the right tree merge to build a unique tree in which all polarities are saturated. There remains a precedence relation which is not fully specified. Specifying it leads to the model shown in Figure 3.

Unlike in LFG or HPSG, in IG, feature structures are not recursive (feature values are atomic). This restriction is partly balanced by the

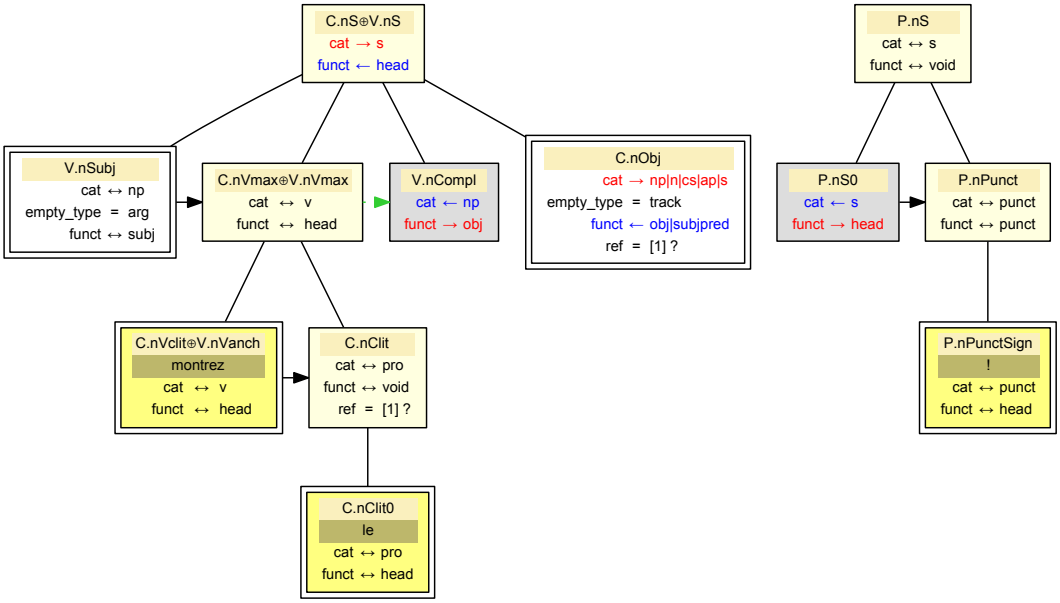


Figure 4: PTD after a first merging step in the parsing of the sentence *montrez-le !*

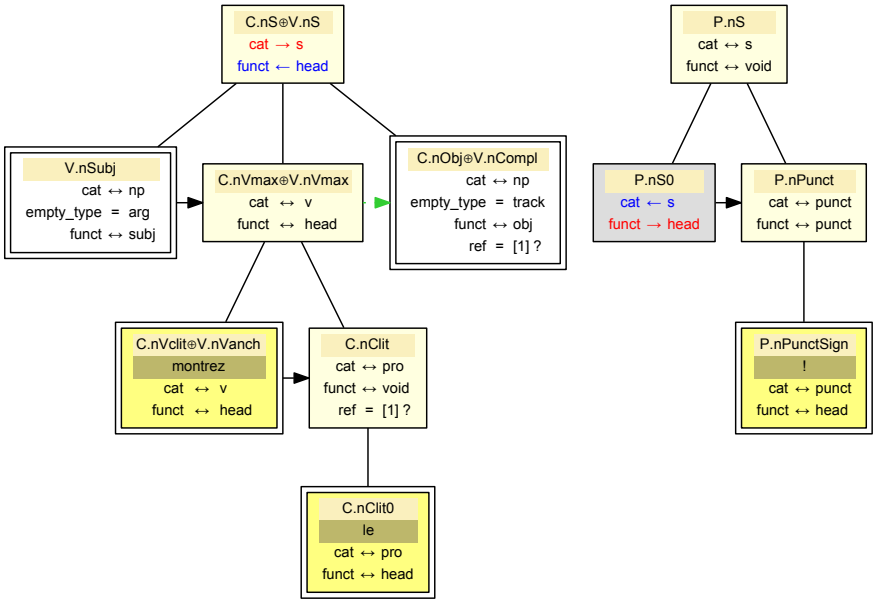


Figure 5: PTD after a second merging step in the parsing of the sentence *montrez-le !*

ability of IG to superpose tree structures. Using virtual polarities to enforce superposition, one can, for instance, impose restriction on the features of a subconstituent as is done by feature equations in LFG.

To summarize, IG combines the strong points of two families of formalisms: the flexibility of *Unification Grammars* and the saturation control of *Categorial Grammars*.

3 THE PRINCIPLES OF THE GRAMMAR FRIGRAM

The formalism of IG is very general and does not impose any linguistic choice apart from the phrase structure tree representation of the syntax of sentences. When specifying a grammar, however, a number of linguistic decisions must be taken such as, for instance, how the notion of a constituent *head* should be defined and whether VP constituents should be used or not. In FRIGRAM parse trees, each node is a constituent linked to a leaf (either an anchor or an empty node) of its subtree called its *head*.

These choices are reflected in the EPTDs of the grammar through *principles*. In this section, we present the principles used in the design of FRIGRAM. These principles are not specific to the French language, however, and can be used or easily adapted to other languages. FRIGRAM, like other grammars with a large coverage, is a large resource and maintaining its consistency is a challenge. The principles facilitate the checking of this consistency. Principles are also heavily used in the conversion from constituent based parse trees to dependency trees discussed in the next section.

The first decision remains the *feature domain*: the choice of the set of feature names and of the possible feature values for each feature name. In FRIGRAM, `cat` is a resource sensitive feature used to encode the category of a constituent.

Principle 1 (cat)**.** *In an EPTD, every node has a* `cat` *feature.*

This feature is used to make a partition on the set of description nodes into two sets: *concrete* and *abstract* nodes.

Definition 1. *A node with a positive or saturated* `cat` *feature is called a* concrete node. *A node with a virtual or negative* `cat` *feature is called an* abstract node.

Another resource sensitive feature called `funct` is used to encode information about lexical heads and grammatical functions.

Principle 2 (funct)**.** *In an EPTD, every concrete node has a* `funct` *feature.*

For lexical head nodes, the special `head` value is used for feature `funct`. For other nodes, the value of the `funct` feature (`subj`, `obj`, ...) encodes the syntactic function of the constituent. There are exceptions (main sentences, moved constituents[5]) and for these exceptions, the `funct` feature carries the special value `void`.

Corollary 1. *From these first two principles, we can infer that:*

1. *each node in a parse tree has a* `cat` *feature;*
2. *each node in a parse tree is the image of exactly one concrete node of the starting PTD;*
3. *each node in a parse tree has a* `funct` *feature.*

Proof. Point 1 follows from the minimality of models: each model node is the image of at least one description node. Point 2 is a consequence of the polarity composition rule: a model node N is either the image of a node with a positive `cat` feature, a node with a negative `cat` feature, and any number of nodes with a virtual `cat` feature or the image of a node with a saturated `cat` feature and any number of nodes with a virtual `cat` feature; in the first (resp. second) case, the node with the positive (resp. saturated) `cat` feature is the only concrete node whose image is the given node N. Point 3 naturally follows from the previous principle and Principle 2. □

The third principle does not refer to linguistic properties but rather to a particular way of attaching linguistic phenomena to words.

Principle 3 (strict lexicalization)**.** *Every EPTD has exactly one anchor node. This anchor node has a saturated* `cat` *feature with an atomic feature value.*

The lexicalization of the grammar is mainly motivated by implementation aspects. Parsing with a lexicalized grammar is guided by

[5] In this case, there are traces of the moved constituents and these traces carry the `funct` feature expressing the syntactic function of the constituent.

the set of lexical units in the input sentences. In IG, as in the TAG formalism, the drawback is that each element of the final tree must be linked to one of the words of the sentence. However, in the case of IG, this is less problematic because of the superposition mechanism which makes it possible to freely split contributions to the final tree among several lexical units.

The last principle relies on the linguistic notions of head and projection which are required to define the concept of *spine*.

Definition 2. *A* spine *in an EPTD is a list of nodes $N_1, N_2, \ldots, N_p$ such that:*

- *for all i such that $1 < i \leq p$, node N_i is a daughter node of N_{i-1};*
- *for all i such that $1 < i \leq p$, node N_i has a saturated feature* `cat` *and a feature* `[funct` $\leftrightarrow$ `head]`*;*
- *node N_1 is a concrete node and its feature* `funct` *has a value different from* `head`*; it is called the* maximal projection *of all nodes belonging to the spine;*
- *node N_p is either an anchor or an empty leaf; in the first case, the spine is called a* main spine*; in the second case, it is called an* empty spine*; in both cases, node N_p is called the* lexical head *of all nodes belonging to the spine.*

Principle 4 (spine)**.** *Every concrete node of an EPTD belongs to exactly one spine.*

This principle means that every concrete node in an EPTD belongs to a continuous chain of concrete projections of a unique head leaf which may be empty.

Corollary 2. *From the strict lexicalization principle and the spine principle we can deduce the following facts:*

1. *every EPTD has exactly one main spine;*
2. *every node N of a tree model has exactly one lexical head in this model, denoted $head(N)$ and defined as follows: the concrete antecedent N_j of N in the initial PTD belongs to exactly one spine $N_1, N_2, \ldots, N_p$ and $head(N)$ is the interpretation in the model of the leaf N_p ending that spine;*
3. *every node N in a tree model which is not a leaf has exactly one daughter node with the feature* `[funct:head]` *(recursively by fol-*

lowing all nodes with feature `[funct:head]`*, we have an equivalent way of finding the lexical head* $head(N)$ *of every node in the model);*

4. *each EPTD node with a positive feature* `cat` *is the maximal projection of some spine.*

Proof. Points 1, 2, and 4 are obvious from definitions and principles. Point 3: N has at least one daughter with the feature `[funct:head]` because its concrete antecedent is on some spine and so it has a daughter node with `[funct:head]`; N has at most one such daughter: by contradiction, suppose that there are two such daughters M and M', then the concrete antecedents N_i of M and N'_j of M' are on two spines $N_1, N_2, \ldots, N_p$ and $N'_1, N'_2, \ldots, N'_q$; hence N is the image of N_{i-1} and N'_{j-1} which are two concrete nodes but two concrete nodes cannot be merged. □

To illustrate the concept of a spine, let us consider the EPTDs of Figure 2. The EPTD associated with the verb *montrez* has two spines: the main spine *V.nS*, *V.nVmax*, *V.nVanch* with its lexical head *V.nVanch*, and an empty spine reduced to a single node *V.nSubj*. In the EPTD associated with the clitic *le*, there are two spines: the main spine *C.nClit*, *C.nClit0*, and an empty spine reduced to node *C.nObj*.

4 THE TRANSFORMATION OF PHRASE STRUCTURE SYNTAX INTO DEPENDENCY SYNTAX

The IG formalism is based on the constituency approach to syntax, as opposed to the dependency approach but the principles introduced in the previous section allow for an automatic transformation of IG constituency parses into dependency structures.

There is a large literature about this kind of transformation (see for example Choi and Palmer 2010). Some of the difficult aspects of this transformation include (i) deciding, for each node of the constituency tree, which of its daughters is the linguistic head and (ii) deciding, for each dependency edge, the grammatical function with which it should be labeled.

In previous versions of FRIGRAM, this transformation relied on the trace of the parsing construction (in IG these traces are the record of the polarities composition but this corresponds to the notion of

derivation tree in TAG). With the most recent version of FRIGRAM, it is not necessary to refer to the parsing construction and the two difficulties mentioned above were taken into account during the grammar building. Thanks to the principles described above, constituency trees contain systematic and precise information about heads of constituents and about their grammatical functions. In consequence, there is a canonical way to derive a dependency tree from the constituency tree.

The transformation is done in two steps. In the first step, all leaves of the model are items of the dependency structure and each node of the constituency tree with a feature `funct` with a value f different from `head` or `void` yields a dependency relation from the head of the mother of the node label with f to its own head. Applied to Figure 1, this gives:

- Node ⟨1.1⟩ produces a dependency relation labeled `subj` from the head of its mother ⟨1.2.1⟩ to leaf ⟨1.1⟩;
- Node ⟨1.3⟩ produces a dependency relation labeled `obj` from the head of its mother ⟨1.2.1⟩ to leaf ⟨1.3⟩;
- Node ⟨2⟩ produces a dependency relation labeled `punct` from the head of its mother ⟨1.2.1⟩ to leaf ⟨2.1⟩.

A relation `ANT` is used to keep track of the link encoded by the `ref` feature between an empty node and the extracted position.

In the second step, to produce more standard dependency structures devoid of ϵ, empty words are removed and their incident dependencies are transferred to their full antecedent, when it exists. Figure 6 (right) shows the effect of empty node removal on the dependency structure obtained above (left).

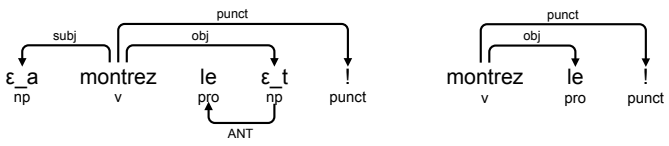


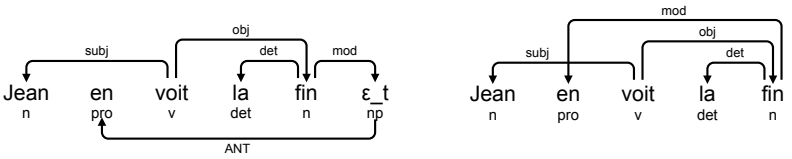
Figure 6: Dependency graphs representing the syntax of the sentence *montrez-le !*

The systematic usage of empty nodes to describe constructions where some constituents are extracted allows for a uniform conversion process. Even for some problematic cases that require non-projective dependency representation like sentence (1), a projective structure is

produced in the first step and the non-projective one is obtained with the second step (see Figure 7).

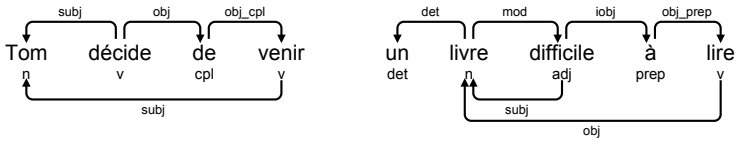
(1) *Jean en voit la fin.*
Jean of it sees the end
'Jean sees the end of it.'

Figure 7: Examples of non-projective dependency structure



In our very simple example, the resulting dependency graph reduces to a tree but in more complex sentences, the dependency graph may contain nodes with several governors or even cycles. In Figure 8, two other examples are given; the first one is a DAG (*Tom décide de venir* 'Tom decides to come'), the second one contains cycles (*un livre difficile à lire* 'a book which is difficult to read').

Figure 8: Examples of dependency structures produced by FRIGRAM



As pointed out in Ivanova *et al.* (2012), different linguistic choice bring incompatible structures. It is the case here, and the structure given by FRIGRAM requires further transformation to match a corpus like Sequoia (Candito and Seddah 2012) for instance.

5 THE ARCHITECTURE OF THE GRAMMAR

A lexicalized grammar for French with a large coverage necessarily has an enormous size: the number of EPTDs in the grammar is the product of the number of entries in the lexicon of inflected words by the average of the number of ETPDs anchored by each inflected word. Now, a lot of these EPTDs are only different at the morphological and phonological level; to factorize these similarities, we use the notion of *EPTD template*. An EPTD template is an EPTD whose anchor is not attached to a particular word. A set of EPTD templates is called an *unanchored grammar*.

The lexicalized grammar is then produced as a combination of an unanchored grammar with a lexicon. Only the much smaller unanchored grammar is stored and the EPTDs are built on the fly during the parsing process. In our case, the unanchored grammar considered is called FRIGRAM.

Another interest of dissociating the lexicon from the grammar is that the lexicon may be written in a way that is totally independent of the grammatical formalism, so that it is reusable with grammars in other formalisms.

This independence of the lexicon with respect to the grammar is also used by FRMG for the formalism of TAG (Villemonte De La Clergerie 2010). It is an advantage with respect to systems in which the lexicon depends more or less strictly on the formalism used for writing the grammar:

- in LKB (Copestake and Flickinger 2000), the lexicon is totally integrated in the typed feature system of the grammar;
- in DotCCG (Baldridge *et al.* 2007), the dependency of the lexicon on the grammar is expressed through the notion of family; a lexical entry is associated with a family, which is a set of syntactic types having a linguistic unity;
- the same notion of family is used in XTAG (XTAG Research Group 2001) for TAG, but here, a family is a set of tree schemas.

5.1 *The modular organisation of the grammar*

It is unthinkable to manually build a grammar with about 4,000 EPTD templates, considering each one individually. Even if it were possible, it would be intractable to maintain the consistency of such a grammar.

Now, the EPTD templates of FRIGRAM share a lot of fragments and it is possible to organize the grammar as a class hierarchy. The structuration of a grammar on the basis of a hierarchy is not new: HPSG uses a hierarchy of typed feature structures via an inheritance relation (Pollard and Sag 1994). Systems where built to help the development of HPSG-based grammars: LKB (Copestake and Flickinger 2000) or TRALE.

IG uses the more generic tool XMG (Crabbé *et al.* 2013). XMG can be used for various formalisms. It was already used for TAG (Crabbé

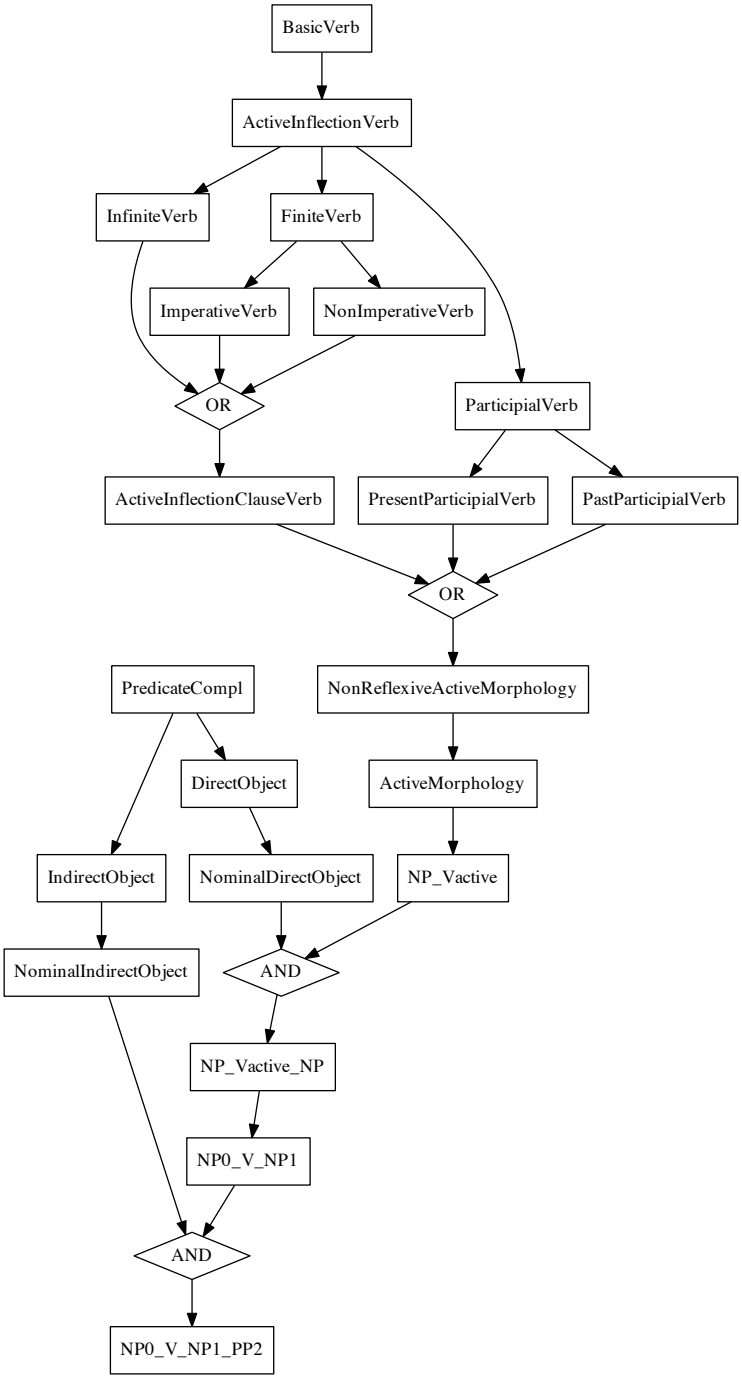


Figure 9: Hierarchy of classes used to define the NP0_V_NP1_PP2 class of transitive verbs with an indirect complement

2005) and IG. XMG provides a language to define a grammar as a set of classes. A class can be defined directly but also from other classes by means of two composition operations: *conjunction* (represented as diamond nodes labeled AND in Figure 9) and *disjunction* (represented as diamond nodes labeled OR).

Each class is structured according to several dimensions. FRIGRAM uses two dimensions: the first one is the syntactic dimension, where objects are EPTD templates, and the second one is the dimension of the interface with the lexicon, where objects are feature structures.

Defining the conjunction of two classes requires a specification of how the components are combined for each dimension: for the syntactic dimension, PTD union is performed; for the dimension of the interface with the lexicon, it is realized as unification between feature structures.

The terminal classes of the hierarchy define the EPTD templates of the grammar that are computed by the XMG compiler. Figure 9 gives the example of a terminal class, the NP0_V_NP1_PP2 class of transitive verbs with an indirect complement, with the hierarchy of classes used to define it. The compiler accumulates the information given by all classes from the top to the bottom of the hierarchy, taking into account the two manners of composing classes. In this way, the compilation of the NP0_V_NP1_PP2 class produces 40 EPTD templates.

For TAG, the compiler also produces tree descriptions, but these are not polarized, and after the compiler, a solver generates the elementary trees that are models of tree descriptions.

The *source grammar* is the set of all classes. In our case, the current source grammar FRIGRAM_S is composed of 425 classes, including 175 terminal ones. The *object grammar* is the set of EPTD templates produced by the compilation of the terminal classes. In our case, the object grammar, FRIGRAM_O, produced from FRIGRAM_S, is composed of 3,890 EPTD templates.[6]

Of course, some general classes can be used in several different contexts. For instance, the classes related to complements of predicative structures are used as subclasses for the classes related to adjectives, nouns, and verbs requiring complements. For the sake of read-

[6] The grammar is systematically described in 280 pages of documentation (Perrier 2014).

ability, the set of classes is organized in a *module hierarchy*. Here is the list of all modules in the alphabetic order, with the number of classes by module between parentheses and a brief characterization of these classes:

- ADJECTIVE (16): adjectives,
- ADVERB (37): adverbs,
- COMPLEMENT (24): complements required by verbs, nouns, or adjectives,
- COMPLEMENTIZER (7): complementizers,[7]
- COORDINATION (12): coordination,
- DETERMINER (12): determiners, except interrogative determiners,
- EXTRACTGRAMWORD (18): extraction (from relative, interrogative, and cleft clauses),
- INTERROGATIVE (17): interrogative pronouns, adverbs, and determiners,
- NOUN (21): common and proper nouns,
- PREPOSITION (13): prepositions,
- PROCLITIC (26): clitic pronouns,
- PRONOUN (21): disjoint pronouns, except interrogative and relative pronouns,
- PUNCTUATION (24): punctuation marks,
- RELATIVE (11): relative pronouns,
- VERB (72): different families of verbs according to their subcategorization frame and specific verbs such as presentatives, modal and causative verbs,
- VERBIMPERSONALDIATHESES (25): different diatheses, active, passive and middle, with an impersonal subject,
- VERBKERNEL (28): classes defining the common verbal kernel of all verbs with the morphology and its interaction with the form of the subject, the syntactic function of the verb, and its voice,
- VERBPERSONALDIATHESES (40): different diatheses, active, passive, and middle, with a personal subject,

[7] The prepositions *à* and *de* introducing direct object infinitives are considered as complementizers, following Huot (1982).

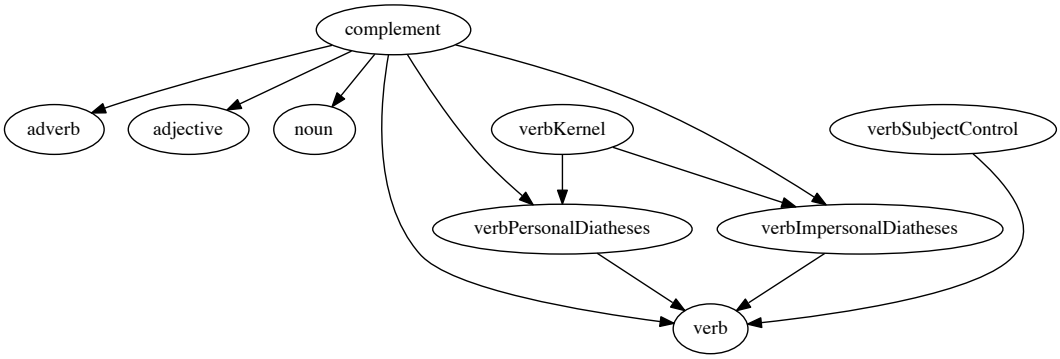


Figure 10: Hierarchy of modules grouping the classes of FRIGRAM$_S$ concerning verbs, nouns, adjectives, and adverbs

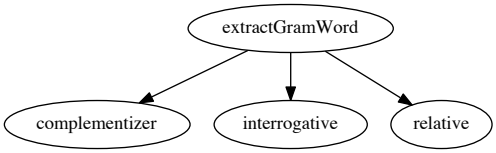


Figure 11: Hierarchy of modules grouping the classes of FRIGRAM$_S$ concerning extraction

- VERBSUBJECTCONTROL (1): control of subjects of infinitives by arguments of the verb governing the infinitive.

The number of classes by module is not proportional to the number of words the module pertains to but also depends on the number and the complexity of the phenomena anchored by these words. For instance, there are few clitic pronouns but they contribute to various syntactic constructions.

Some classes of one module are defined from classes of another module. We can represent this property with a graph where an edge means that some classes of the target module are defined from classes of the source module. Figure 10 shows these dependencies for the modules concerning verbs, nouns, and adjectives.

Figure 11 shows these dependencies for the modules modeling extraction from relative, interrogative, and cleft clauses. The modules

absent in Figure 10 and Figure 11 are isolated ones, without external dependencies.

A natural question arises: since XMG can be used for various formalisms, TAG and IG in particular, is it possible to re-use a grammar constructed in one formalism to build a new grammar in another formalism? The problem is that the organization of a grammar constructed with XMG is very close to the formalism, especially to the operation modeling syntactic composition. In TAG, this operation is adjunction, so that grammatical information tends to be anchored at verbs. In IG, this operation is merging of PTDs, which is more flexible, so that it is possible to anchor information at grammatical words.

5.2 *The link with a lexicon independent of the formalism*

The full grammar is produced from FRIGRAM$_O$ and a lexicon. Each EPTD template from FRIGRAM$_O$ is associated with a feature structure, called its *interface*, which describes a syntactic frame corresponding to lexical units able to anchor it. Lexicon entries are also described through feature structures. The set of features used in the interfaces differs from the one used in EPTDs because they do not play the same role: they do not aim at describing syntactic structures but are used for describing the morpho-syntactic properties of the words of the language anchoring the EPTDs in a way independent of the formalism.

Figure 12 shows an EPTD generated by the NP0_V_NP1_PP2 class with its interface above. The interface appears as a two-level feature structure. Features at the first level give the constituents of the frame associated with the EPTD. In our example, the `head` feature represents the verb. The `subj`, `obj`, and `iobj1` features respectively represent the subject, the direct object and the indirect object of the verb.

The second level describes the properties of each constituent of the frame. For instance, the value of the `head` feature is a feature structure describing some morpho-syntactic properties of the verb. Among the features present in this structure, `[impers:maybe|never]` says that the verb is either a verb that accepts a personal and impersonal construction of its subject or a verb with only a personal construction. The feature `[pronominal:maybe|never]` says that the verb may accept a reflexive object or it does not admit a reflexive clitic.

In the lexicon, the entries are pairs of an inflected word and feature structure. The feature structure has exactly the same format as

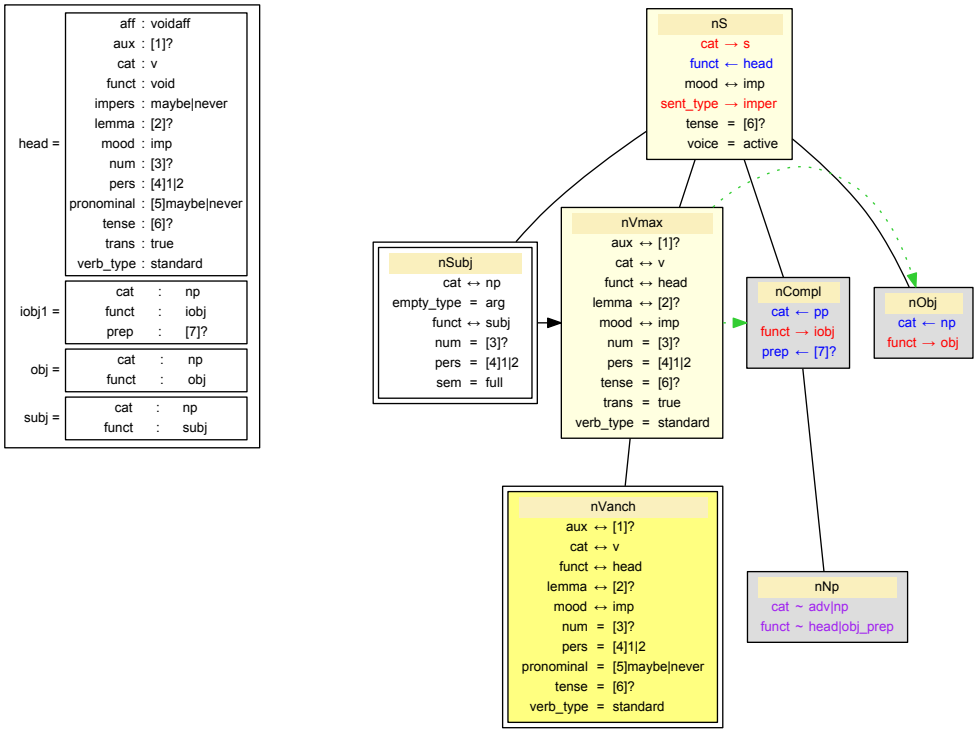


Figure 12: One of the EPTDs generated by the NP0_V_NP1_PP2 class

the interfaces of the grammar. For instance, Figure 13 shows a lexical entry for the verb *montrez* used with a direct and indirect object, like in the sentence *montrez-le moi !* 'show it to me!'.[8]

The anchoring of an EPTD template is performed by unifying its interface with the entries of the lexicon. At the first level, interfaces and lexical entries are viewed as closed structures: they unify if they have exactly the same set of features. At the second level, they are viewed as open structures: for a given feature of the first level, its value in the interface unifies with its value in the lexicon entry in the sense usually given to the operation of unification, which may entail the addition of second level features to the interface not present initially. For instance, the interface of the EPTD template presented in

[8] The question mark after the name of feature `iobj1` expresses that this feature is optional.

Figure 13:
Entry of the verb *montrez* in the lexicon

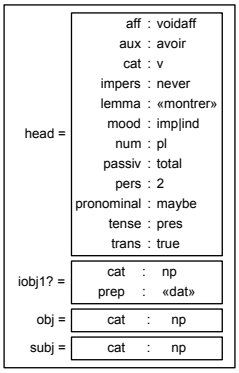


Figure 12 succeeds in the unification with the lexical entry presented in Figure 13. At the first level, they have exactly the same set of features: `head`, `iobj1`, `obj`, and `subj`. At the second level, the values of the first level features unify in a standard way.

Since there is a co-indexation between features of the EPTD template and features of the interface, some feature values of the EPTD may be instantiated during anchoring. Figure 14 shows the anchored EPTD resulting from the unification between the interface of the EPTD template from Figure 12 and the lexical entry from Figure 13. As a side effect of anchoring the values of features `aux`, `lemma`, `num`, `pers`, `tense`, and `prep` have been instantiated.

The lexicon used to anchor FRIGRAM$_O$ (called FRILEX) combines morphological information extracted from ABU[9] and from Morphalou (Romary *et al.* 2004) with syntactic information for verbs extracted from Dicovalence (Van den Eynde and Mertens 2003). FRILEX contains 530,000 entries. To avoid size explosion, the required EPTDs of the grammar are built on the fly during parsing.

6 THE COMPANION PROPERTY AND THE CONSISTENCY OF THE GRAMMAR

Our ambition is to build a large coverage grammar for French syntax. Even if the hierarchical structure of the grammar makes it more compact and facilitates maintenance, the size of the grammar may be

[9] `http://abu.cnam.fr/DICO/mots-communs.html`

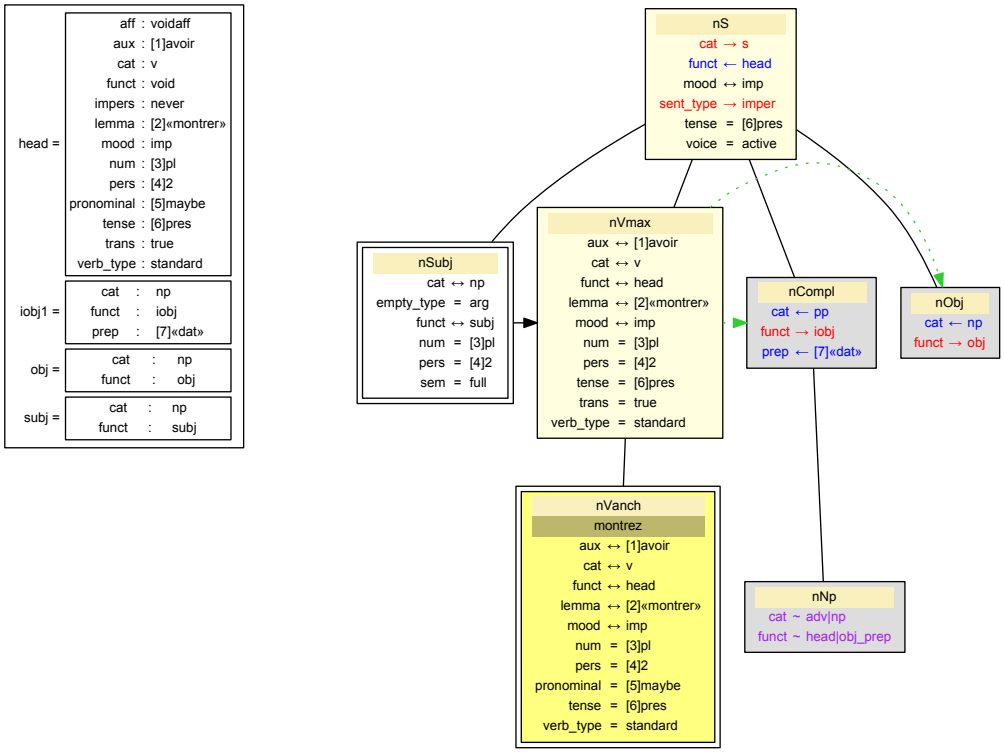


Figure 14: Anchored EPTD for the verb *montrez*

important and global consistency is difficult to maintain. The problem is that the different classes of the grammar source, even if they are not linked in the hierarchy, are generally not independent because EPTDs interact through the process of syntactic composition. For instance, in the parsing of the sentence *montrez-le !*, the EPTD anchored by the verb *montrez* interacts with the EPTD anchored by the clitic pronoun *le*, though the corresponding classes in the source grammar are totally independent. It is essential to check that this interaction works correctly.

A standard way to check global consistency of the grammar is to parse real sentences from test suites or corpora. However, the IG formalism provides another mechanism, complementary to the parsing of real sentences, to help checking the consistency of a grammar in a static way based on the EPTDs of the grammar without using

parsing. This mechanism uses the *Companion Property*. Originally, this property was introduced by Bonfante *et al.* (2009) to perform lexical disambiguation with IG.

Let us consider an interaction grammar.

Definition 3. *A companion of an unsaturated polarized feature in an EPTD of the grammar is another polarized feature of an EPTD such that the first feature is saturated by the second one: a merging of their respective nodes leads to a consistent PTD.*

A PTD is called *consistent* when it does not contain any unsatisfiable constraint; in other words, there exists some context in which the PTD can produce a model.

For instance, consider the EPTD associated with the verb *montrez* in Figure 2. A companion of the positive feature [funct → obj] is the negative feature [funct ← obj|subjpred] of the EPTD associated with the clitic pronoun *le*. The notion of companion can be expressed at the template level: if a template EPTD contains an unsaturated polarized feature, it requires to be saturated by some other complementary feature and we can search for companions (polarized features in a template EPTD) that are able to saturate it. Thanks to the medium size of FRIGRAM$_O$, it is possible to exhaustively compute the set of companions of all polarized features of the EPTDs templates of grammar FRIGRAM$_O$.

For instance, consider the EPTD template (noted E_0) corresponding to the EPTD anchored with *montrez* in Figure 2. It contains a positive feature [funct → obj]; scanning FRIGRAM$_O$, we find 107 companions. Moreover, for 95 companions (out of the 107), we know that they necessarily are companions coming from EPTD templates for which the anchor is on the right of the anchor of E_0 after merging. The remaining 12 cases do not imply any order constraint on anchors.

As already said, companions are used for lexical disambiguation: when parsing a sentence, if an unsaturated feature of an EPTD E fails to find a companion in the EPTDs of the other lexical units of the sentence, E cannot be used in the parsing and so it can be removed. But companions can also be useful to help grammar development. The first diagnostic is when a polarized feature has an empty companion set: this means that the corresponding EPTD cannot be used in any grammatical parse; in this case this EPTD can be removed from the grammar

or there is some mistake in its definition. Other kinds of observations can be used by the expert: if a polarized feature has companions only on the left or on the right, it can be checked whether this corresponds to linguistic intuition. The full grammar contains 56,462 unsaturated polarities waiting for a companion: 21.2% have companions only on the right and 11.1% have companions only on the left.

During the FRIGRAM development, many inconsistencies were discovered using information about companion sets. In particular, some verbs requiring complements were found to have no companion because of the incompatibility between features or polarities of the expected complements and those of the EPTDs attached at the potential complements. It greatly helped to correct more or less deep defects and errors of FRIGRAM.

7 COVERAGE OF THE GRAMMAR

FRIGRAM covers a lot of syntactic phenomena: the different verb diatheses (active, passive, middle, causative, and impersonal diatheses), raising and control verbs, different types of sentences (declarative, interrogative, exclamative, imperative), extraction from interrogative, relative, cleft and dislocated clauses, noun complement clitic pronouns, quantifier pronouns, tough movement, coordination, and others. Within the limits of this article, it is not possible to describe all phenomena covered by FRIGRAM in an exhaustive way, but the reader can find a more complete information in the documentation of FRIGRAM (Perrier 2014).

In this section, we have chosen to present some of the most complex syntactic phenomena in French: causative constructions, negation, comparative, and consecutive constructions, extraction with pied-piping. Some of these have been little studied, especially comparative and consecutive constructions.

The modeling of these phenomena is constrained on the one hand by the formalism of IG and on the other hand by the rules of the French grammar. For the latter, our guide is the grammar of Riegel *et al.* (1999).

For every phenomenon, we compare the modeling in IG with the modeling in other formalisms, but most publications presenting such works are theoretical without any implementation and we know that

from a theoretical idea to its implementation in a grammar with a large coverage there is a long way which may be fraught with pitfalls (Bender 2008).

7.1 *Causative constructions*

In a causative construction, a causative verb (*faire* or *laisser*[10] in French) combines with an infinitive in the active voice. Here are examples illustrating this construction. For every sentence, the causative auxiliary and the complement infinitive are in bold.

(2) *Jean le **fait** **remplir**.*
Jean it makes fill
'Jean makes someone fill it.'

(3) *Jean **fait** se **rencontrer** les ingénieurs aujourd'hui.*
Jean makes meet the engineers today
'Jean makes the engineers meet today.'

(4) *Jean s' est **fait** **contrôler**.*
Jean himself has made control
'Jean has been controlled.'

(5) *Que Marie mange beaucoup la **fait** **dormir**.*
That Marie eats a lot her makes sleep
'That Marie eats a lot makes her sleep.'

(6) *Jean **fait** **prendre** par Marie son billet de train.*
Jean asks to take by Marie his ticket of train
'Jean asks Marie to take his train ticket.'

(7) *Jean **fait** **balayer** la cour à Marie.*
Jean asks to sweep the yard to Marie
'Jean asks Marie to sweep the yard.'

All these sentences are parsed with FRIGRAM. Sentence (2) illustrates clitic climbing in causative constructions: the clitic pronoun *le* is the direct object of *remplir* but it is attached at the causative verb *fait*.

Sentence (3) shows that clitic climbing is not performed if the clitic is a reflexive pronoun, *se* in the example that refers to the object *les ingénieurs* in the example.

However, in Sentence (4), the reflexive pronoun *se* refers to the subject of the causative verb *Jean*, while being the object of *contrôler*.

[10] The verb *laisser* can be only partially considered as a causative verb (Abeillé *et al.* 1997).

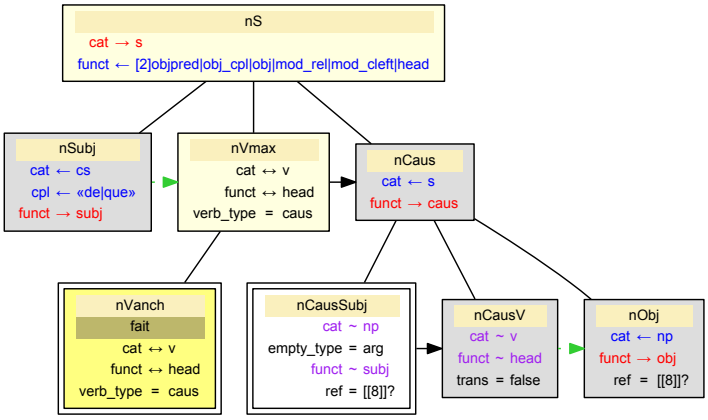


Figure 15:
Anchored EPTD for the verb *fait* used as a causative verb with a specific object representing the subject of an intransitive caused verb

In this case, it climbs. Moreover, in this example, if the construction is a causative from a syntactic point of view, it is not the case from a semantic point of view, because *Jean* is not the causer of the action *contrôler*.

Sentence (5) shows a sentence where the causer is a complete clause: *Que Marie mange beaucoup.*

Sentences (6) and (7) illustrate the following rule: if the caused verb is transitive, its subject is expressed as an agent complement or indirect object.

In FRIGRAM, causative verbs are considered as special full verbs, as Figure 15 illustrates. The EPTD, shown in this figure, is attached at the verb *fait* used as a causative verb, with an intransitive caused verb. The caused verb is the head of an infinitive represented with the node *nCaus*. A specific complement, representing the subject of the caused verb introduced by the causative verb is considered a sub-constituent of the infinitive clause headed by the caused verb. In the EPTD, it is represented by the node *nObj* with the function `object`. A co-indexed feature `ref` indicates that it refers to the same entity as the empty subject of the infinitive represented by the node *nCausSubj*.

The question arises: why have we chosen to put this node *nObj* as a sub-constituent of *nCaus* and not of *nS*? The reason is that it is possible to insert specific complements linked to the causative construction between the caused verb and its own complements. Sentence (6) illustrates this problem: the agent complement *par Marie*, depending on the causative construction, comes between the caused verb *prendre* and its

object *son billet de train*. As a consequence, it must be put in the same constituent.

An alternative way of modeling causative constructions would be to consider causative verbs as auxiliaries, like tense or passive auxiliaries. It would require adding a specific entry in the grammar for all infinitives likely to take a causative auxiliary, which would increase the size of the grammar and lexical ambiguity in parsing. Another drawback comes from the flat representation that this entails. Let us consider Sentence (3). If we consider *fait* as a usual auxiliary of the verb *rencontrer*, node *nS* representing the whole sentence will have two daughter nodes as sub-constituents with the same function `object`: the own object of *rencontrer*, the clitic pronoun *se*, represented by its trace, and a specific object, *les ingénieurs*, introduced by the causative construction. Thus, if we want to attach every object at its verb, we need additional information to know which object is attached at which verb. We have the same problems for dative complements: in a flat structure, we can have two dative complements at the same level, one that is attached at the causative verb and another one that is attached at the infinitive.

Now, how to model constraints on clitic climbing, as they are expressed in the previous examples? In order to limit the ambiguity of the grammar, they are not attached at the causative verb but at the clitic pronouns. Figure 16 shows an EPTD for a reflexive clitic that is an object of the caused verb and that refers to the subject of the causative verb. Such a clitic must climb to the causative verb, as Sentence(4) illustrates. A node *nConst* represents the trace of this object at the canonical object position in the clause *nS0* headed by the caused verb, *contrôler* in our example. The clitic *se*, represented by the node *nClit*, is attached at the auxiliary *est* of the causative verb *fait*, represented by the node *nVclit*. Finally, the feature `[ref=[[8]]?]` indicates a co-reference with the subject *Jean*. There is another lexical entry for *se* when it does not climb in a causative construction, as Sentence (3) illustrates. For non-reflexive clitic pronouns, there is only one lexical entry which forces climbing.

There are in-depth studies of French causative constructions in two other formalisms, HPSG and LFG. In all studies, the discussion is about the choice between the flat and the biclausal representation, which relates to a linguistic choice: to see a causative verb as any

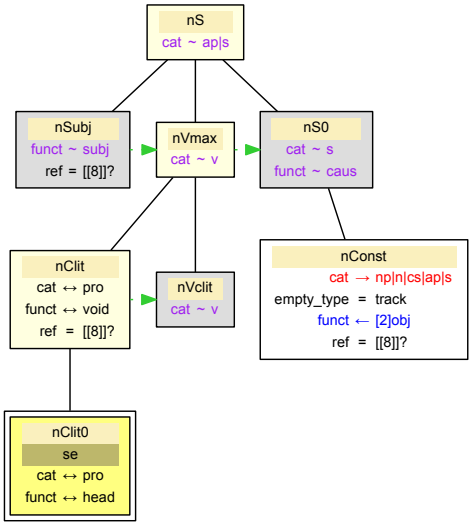


Figure 16:
Anchored EPTD for the clitic pronoun *se* used as an object of a caused verb referring to the subject of the causative verb

control verb or to see it as constituting a complex verb with its object infinitive.

Abeillé *et al.* (1997) do not choose any of the options but dedicate a specific representation to each case: the biclausal representation is used when the causee is an object clitic and when there is no clitic climbing; the flat representation is used in other cases. The two cases are not completely disjunct, so that the two representations can be used when the caused verb is intransitive and the causee is an object clitic as in Sentence (5), whereas FRIGRAM provides a unique parse in this case. On the other hand, Abeillé *et al.* (1997) accept very rare constructions which are rejected by FRIGRAM. Here are examples illustrating these cases.

(8) *Jean le **fait** **remplir** la citerne.*
Jean him makes fill the cistern
'Jean makes him fill the cistern.'

(9) *Jean le **fait** lui **téléphoner**.*
Jean him makes him call
'Jean makes him call him.'

In Sentence (8), the causee is an object clitic, though the caused verb has a direct object. In Sentence (9), the clitic *lui* has not climbed. It is

possible to take these cases into account in FRIGRAM, but the interest is not obvious, because of the rarity of their occurrences.[11]

Both in FRIGRAM and in the proposal of Abeillé *et al.* (1997), not all the constraints on the cliticization of the arguments for the causative verb and infinitive are taken into account. For instance, the following ungrammatical sentence, taken from Yates (2002), is accepted.

(10) ∗*Pierre* *lui* ***a fait*** ***téléphoner*** *Marie*
Pierre him made call Marie
'Pierre made Marie call him.'

Yates (2002) uses the potentiality of LFG to represent the complex constraints on cliticization in causative constructions illustrated by Sentence (10). At the constituent level, the c-structure expresses a flat representation, whereas at the functional level, the f-structure allows a sharing between arguments of the infinitive and the causative verb. He shows that his ideas can be transposed in HPSG.

To transpose them in IG would entail substantial changes in the grammar. Putting the caused verb and its complement in a flat structure at the same level as the causative verb requires a specific entry in the grammar for the caused verb. The usual entries for infinitives no longer work because the mood of the clause is given by the causative verb. This addition of a new entry must be repeated for all subcategorization frames of infinitives at the active voice. This would increase the size and ambiguity of the grammar.

7.2 *Negation*

In French, negation is most often expressed with the clitic *ne* paired with a negative grammatical word which can be an adverb (*pas*, *guère*...), a pronoun (*personne*, *nul*...), or a determiner (*aucun*...). To name all these words in a unique manner, we use a non-standard term, *negative satellites*, which expresses that the words must be paired with *ne*. The following examples illustrate different cases of negative satellites. The clitic *ne* and its satellites are in bold.

(11) *Jean* **ne** *mange* **pas** *de* *pommes.*
Jean eats not apples
'Jean does not eat apples.'

[11] No occurrence of these constructions exists either in Sequoia or in the FTB.

(12) *Marie* **ne** *pense connaître la femme d'* **aucun** *ingénieur.*
Marie think to know the wife of any engineer
'Marie thinks to know the wife of no engineer.'

(13) *Jean* **ne** *travaille avec l' appui de* **personne.**
Jean works with the support of nobody
'Jean works with the support of nobody.'

(14) *Jean* **ne** *pense pouvoir travailler* **que** *dans sa chambre.*
Jean thinks to be able to work only in his room
'Jean thinks to be able to work only in his room.'

The pairing of *ne* with one negative satellite is expressed in FRIGRAM with a polarized feature `neg` which is attached at the clause constituting the scope of the negation. Particle *ne* provides the positive feature `[neg →true]` to neutralize the dual negative feature `[neg ←true]` given by the negative satellite.

Figure 17 shows the EPTD attached at the clitic *ne* on its left. The node *nS* represents the sentence that is the scope of the negation. It carries the positive feature `[neg →true]`. The particle *ne* appears as a clitic put before the verb represented by the node *nV*. The maximal projection of the clitic represented by the node *nAdvmax* cannot receive any modifiers, which is indicated by the fact that the node is closed (double rectangle on the figure).

A difficulty comes from the fact that the negative satellite paired with *ne* can be situated in a constituent that is embedded more or less deeply in the clause that is the scope of the negation. This property applies only to some satellites. In particular, it is true for *aucun* and *personne*, as Sentences (12) and (13) show it. The satellites are situated in noun phrases: *aucun ingénieur* in Sentence (12) and *personne* in Sentence (13). If these noun phrases are noun complements, as in the examples, they can be embedded more or less deeply in a clause and, moreover, this clause can itself be embedded in the clause that is the scope of the negation.

That is the case for Sentence (12): from the satellite word *aucun* to the scope of the negation, there is chain of constituents: a noun phrase *aucun ingénieur*, a prepositional phrase *d'aucun ingénieur*, another noun phrase *la femme d'aucun ingénieur*, and an infinitive *connaître la femme d'aucun ingénieur*.

Figure 17 (on the right) shows the EPTD anchored by *aucun* and used in the parsing of Sentence (12). In this EPTD, the node *nS* repre-

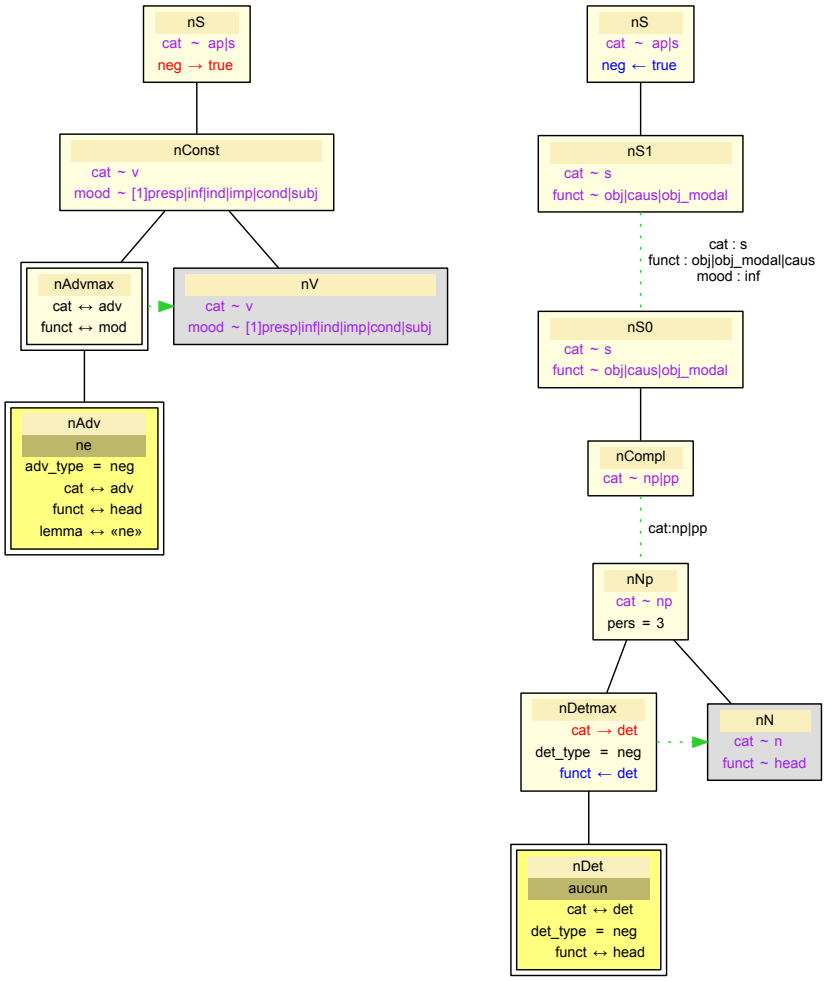


Figure 17: EPTD attached at the clitic *ne* and an EPTD attached at the negative satellite *aucun*

sents the clause that is the scope of the negation. It is labeled with the negative feature [neg ← true].

The node *nS1* represents the immediate sub-constituent of *nS* that is the clause including *aucun*. The clause *nS0* that is the location of *aucun* can be embedded more or less deeply in *nS1*, which is expressed with an underspecified dominance relation from *nS1* to *nS0*. In Sentence (12), the nodes *nS0* and *nS1* are merged and they represent the clause *connaître la femme d'aucun ingénieur*.

The node *nCompl* represents the complement that is a sub-constituent of *nS0* and that is the location of *aucun*, *la femme d'aucun ingénieur* in the example. The node *nNp* represents the noun phrase that is the location of *aucun*, *aucun ingénieur* in the example. Another underspecified dominance relation from *nCompl* to *nNp* expresses the possibility for *nNp* to be embedded more or less deeply in *nCompl*.

Now, even if the location of a satellite word like *aucun* is relatively free, it is not completely free. Here are examples of unacceptable sentences in French.

(15) ∗*Jean* **ne** *vient* *parce qu'* *il* *connaît* **aucun** *invité.*
Jean is coming because he knows no guest
'Jean does not comes because he doesn't knows any guest.' (intended)

(16) ∗*Marie* **ne** *pense* *connaître* *une* *fille* *qui* *ait* **aucun** *défaut.*
Marie thinks to know a girl who has no defect
'Marie does not think that she knows a girl without any default.' (intended)

Sentence (15) illustrates the fact that *aucun* cannot be located inside an adjunct clause depending on the clause including the particle *ne*. In IG, such a constraint is expressed on underspecified dominance relations with feature structures. In the EPTD on the right of Figure 17, the following feature structure labels the underspecified dominance relation from *nS1* to *nS0*: `{[cat:s], [funct:obj|obj_modal|caus], [mood:inf]}`. This feature structure means that each node of the model that is between the nodes *nS1* and *nS0* must be labeled with `[cat:s]`, `[mood:inf]`, and `[funct:obj]` or `[funct:obj_modal]` or `[funct:caus]`. In other words, the clause where *aucun* is located must be embedded in a sequence of infinitive clauses that are object of transitive verbs, causative verbs, or modal auxiliaries.

Sentence (16) illustrates a constraint on another underspecified dominance relation in the EPTD of Figure 17 on the right: from *nCompl* to *nNp*. This relation is constrained by the feature structure `{[cat:np|pp]}`. Such a constraint means that between *nCompl* and *nNp* there may be only a sequence of noun or prepositional phrases. This entails the failure of parsing Sentence (16). In this sentence, from *nNp* to *nCompl* there is the following sequence of constituents: a noun phrase *aucun défaut*, a relative clause, *qui ait aucun défaut*, and another noun phrase, *une femme qui ait aucun défaut*. The second constituent of this sequence violates the constraint on the underspecified dominance relation from *nCompl* to *nNp*.

Sentences (13) and (14) show that *personne* and *que* behave in a similar manner as *aucun*, which is modeled in FRIGRAM with a similar EPTD. For *que*, there is a difference: there is only one underspecified dominance relation, between *nS1* and *nS0*.

Another difficulty comes from the fact that one clitic *ne* can be paired with several negative satellites, as Sentence (17) shows. In this sentence, there are two satellite adverbs: *pas* and *que*.

(17) *Jean* **ne** *mange* **pas** **que** *des* *pommes.*
Jean does not eat only apples
'Jean does not eat only apples.'

In the IG formalism, one positive feature [neg → true] must be saturated by exactly one negative feature [neg ← true]. Our solution is to distinguish between the main and secondary satellite word. The main satellite word brings the negative feature [neg ← true] and the secondary satellite word brings a virtual feature [neg ∼ true].

All negative satellites can play the role of main satellites. How should the negative satellites that can also play the role of secondary satellites be determined? We have chosen a pragmatic criterion: if a negative satellite can occur simultaneously with a lot of other negative satellites, we consider it a potential secondary satellite. It is not completely satisfactory because it ignores some cases. For instance, *que* can co-occur with *rien*, *guère*, *jamais*, *pas*, *plus*, whereas *pas* can only co-occur with *que*. As a consequence, *que* is considered a secondary satellite, while *pas* is considered only a main satellite.

Regarding the specific studies about the formalization of the negation syntax in French, we propose to start the comparison with work that is related to the study of the phenomenon from a strictly syntactic point of view. Following Abeillé and Godard (1997), Kim and Sag (2002) propose to model French and English negative adverbs in the framework of HPSG. For French, they restrict the study to the adverbs behaving as *pas*, that is with a very constrained position: *guère*, *jamais*, *plus*... They do not consider *que* which has a more free position.

Their work focuses on the possible positions of the negative adverb in the constituent tree of the sentence, according to the mood of the head verb. They conclude that with non-finite verbs it is a preverbal VP-adjunct, and with finite verbs it is taken as a postverbal complement sister of the other complements in the VP. *Ne pas* put be-

fore an infinitive is considered a VP-modifier. *Ne* put before a finite verb is considered an affix and a lexical rule transforms the lexical entry of the finite verb into a negated entry where a slot for the negative adverb is added to the subcategorization frame of the verb. In this way, the number of negation adverbs that it is possible to put after a finite verb is controlled by the subcategorization frame.

In IG, the notion of a verb phrase does not exist, so that negative adverbs are considered verb modifiers. The constraints about the linear order between the verb and negative adverb according to the mood of the verb are expressed with two different EPTDs corresponding to the two positions. For the preverbal position, the modeling of the negative adverb as a verb modifier entails a small limitation: it is not possible to reflect wide scope of *ne pas* over a conjunction of coordination, as in the following example extracted from Abeillé and Godard (1997).

(18) *Paul promettait de* **ne pas** *lire le journal ou regarder la télévision*
Paul promised not to read the newspaper or watch television.
'Paul promised not to read the newspaper or watch television.'

To take this phenomenon into account, it is sufficient to add a unique EPTD anchored by *ne pas* which has the function of infinitive modifier besides the EPTDs used for *ne* and *pas* with other moods.

In the restricted context chosen by Kim and Sag (2002), the dependency between the clitic *ne* and the negative satellite is relatively simple. Godard (2004) studies this dependency in depth in a general context, where it is more complex. Moreover, she also considers the semantic dimension: the satellite is considered as a quantifier and the clitic *ne* marks the scope of this quantifier. She proposes a model of the dependency in the framework of HPSG. This dependency is distant. Whereas IG uses its system of polarities combined with the relation of underspecified dominance to express distant dependencies, HPSG uses the propagation of features in the syntactic tree from node to node.

7.3 *Comparative and consecutive constructions*

Some adverbs, while acting as modifiers, are correlated with conjunctions or prepositions introducing a clause in a comparative or consec-

utive construction, as the following examples show. The adverbs and their correlated grammatical words are in bold.

(19) *Il connaît les parents de* **trop** *d'élèves* **pour** *ne pas venir.*
He knows the parents of too many students to not come
'He knows the parents of too many students to not come.'

(20) *Jean a* **tellement** *travaillé* **qu'***il peut se reposer.*
Jean has so much worked that he may have a rest
'Jean has worked so much that he may have a rest.'

(21) *Le paysage est* **plus** *ensoleillé* **qu'** *il ne l' est en hiver.*
The landscape is more sunny than it is in winter
'The landscape is more sunny than it is in winter.'

(22) *Le paysage est* **plus** *ensoleillé maintenant* **qu'** *en hiver.*
The landscape is more sunny now than in winter
'The landscape is more sunny now than in winter.'

The first two examples illustrate the consecutive construction and the last two illustrate the comparative construction. Like negation, the two constructions use a correlation between two distant grammatical words. This correlation was analyzed from a linguistic point of view either in general French grammars (Grevisse and Goosse 2008; Riegel *et al.* 1999), or in specific studies for the comparative construction (Fuchs *et al.* 2008), but, to our knowledge, there is no specific study of their modeling in grammatical formalisms.

As for negation, the modeling in IG of the correlation between the adverb and conjunction in both constructions uses the system of polarities. We propose to develop on how the comparative construction of Sentence (22) is modeled within FRIGRAM.

Figure 18 represents the EPTDs used for *plus* and *que* in the parsing of Sentence (22). The correlation between the two words is expressed with polarized features, here the features `cat`, `funct`, and `sent_type` of the nodes *nCompl0* and *nCs*.

In the EPTD of *plus*, the node *nConst* represents the word after its modification by *plus*, *plus ensoleillé* in our example. The node *nC* represents the expression that is the scope of the construction; in our example, it is the adjectival phrase *plus ensoleillé maintenant qu'en hiver*.

In the EPTD of *que*, the node *nCs* represents the clause complemented by *que* which is an argument of the adverb triggering the comparison. The node *nS* represents the clause without its complementizer. In our example, as in most cases, the clause includes an ellipsis.

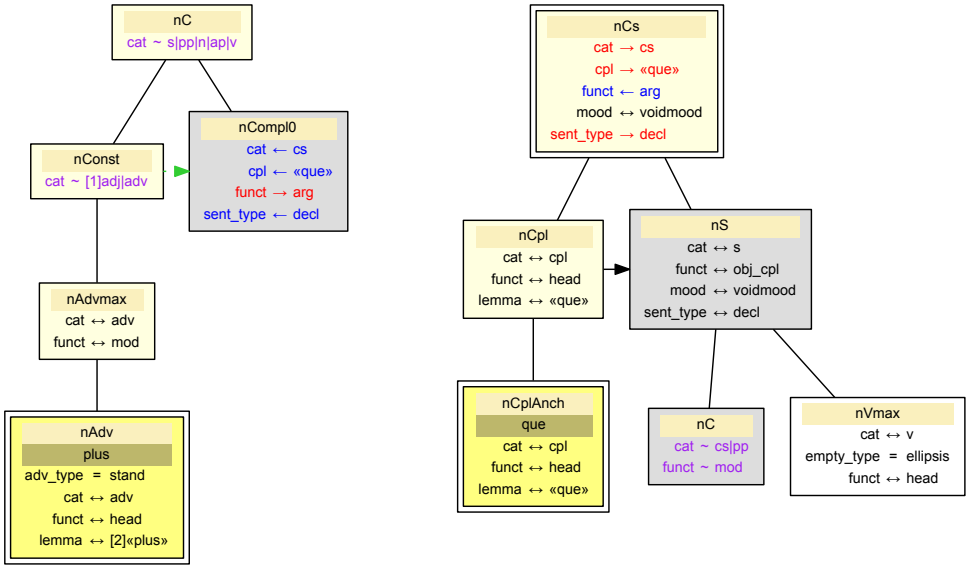


Figure 18: EPTDs anchored by *plus* and *que* for the parsing of Sentence (22)

It reduces to the prepositional phrase *en hiver* and the complete clause would be *il ne l'est en hiver*. The elided verb is represented by the node *nVmax* and the node *nC* represents the complement *en hiver*.

The modeling is similar for the other examples, except for Sentence (19), where *trop d'élèves* is embedded in a constituent which is not at the same level as the clause expressing the consequence *pour ne pas venir*. In FRIGRAM, it is expressed in the EPTD of *trop* with an underspecified dominance relation from the constituent representing the scope of the construction, the whole sentence in our example, to the noun phrase determined by *trop de*, *trop d'élèves* (see Figure 19).

7.4 *Extraction with pied-piping*

All relative, cleft, and interrogative clauses with partial interrogation give rise to extraction of constituents. These constituents are put at the beginning of the clause from which they are extracted, and in our approach an empty constituent remains at the initial place as a trace.

Extraction is one of the syntactic phenomena in French that are the most difficult to formalize because it interacts with other phenomena, especially subject inversion and pied-piping. In the limits of this article we propose to focus on pied-piping.

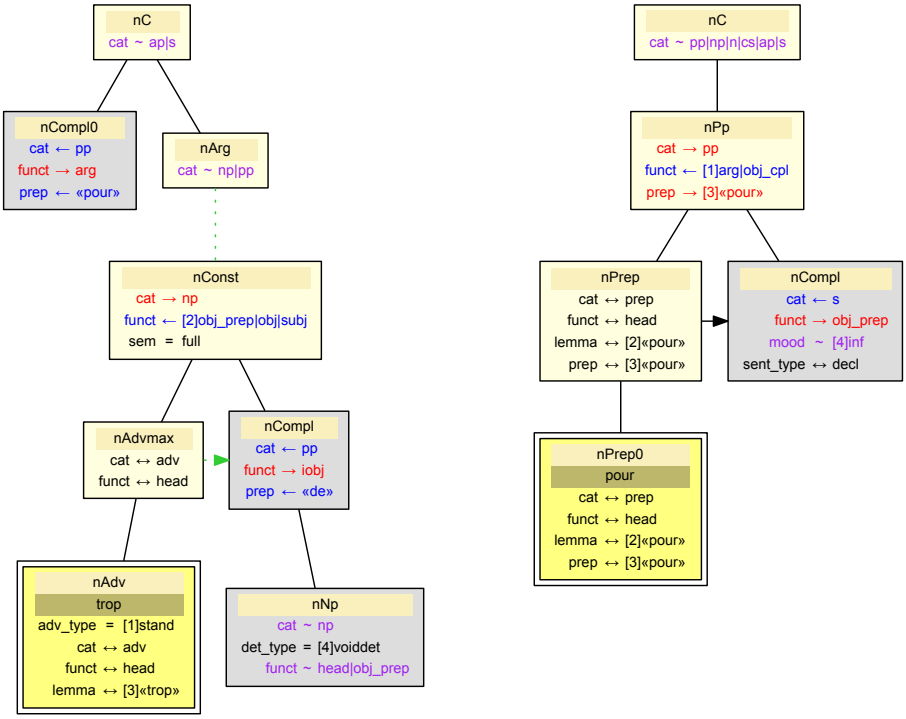


Figure 19: EPTDs anchored by *trop* and *pour* for the parsing of Sentence (19)

The three sentences below illustrate the phenomenon. In each example, the extracted constituent is put in square brackets and the grammatical word triggering extraction, called wh-word in the following, is displayed in bold type. The trace of the extracted constituent is marked with the symbol □.

(23) *Marie connaît Jean [dans l' entreprise de* **qui***] elle pense*
Mary knows John in the company of whom she believes
travailler bientôt □.
to work soon
'Mary knows John, in whose company she believes to work soon.'

(24) *[Au patron de* **quelle** *entreprise] Sue veut -elle parler* □ *?*
to the boss of which company Sue wants to-speak
'Which company does Sue want to speak to the director of?'

(25) *[Au directeur de* **laquelle***] Marie veut -elle parler* □ *?*
to the director of which-one Mary wants to-speak
'Which one does Mary want to speak to the director of?'

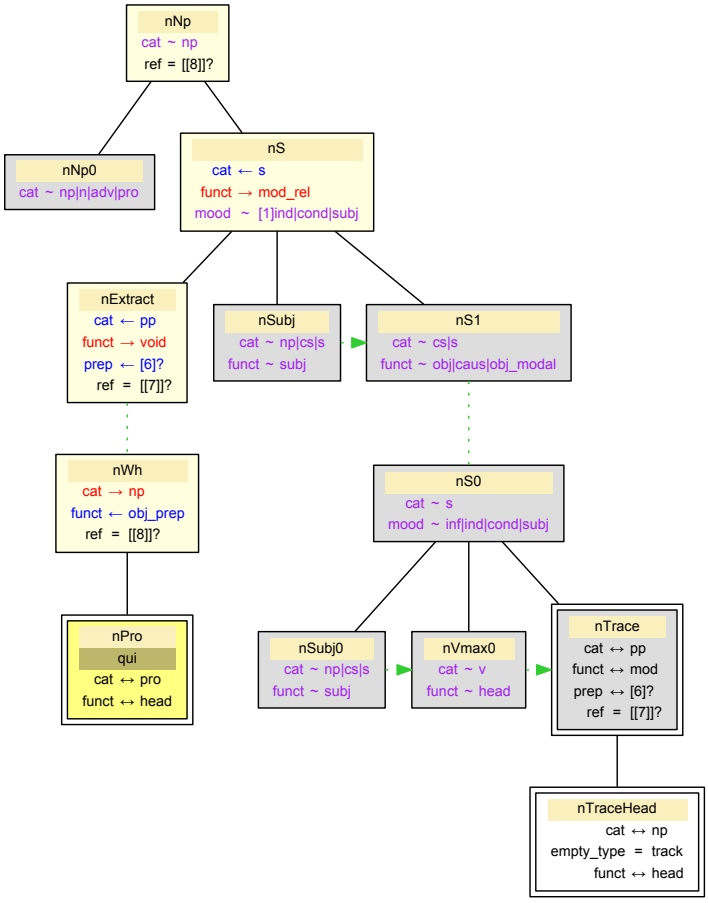


Figure 20: EPTD anchored by the relative pronoun *qui* and used in the parsing of Sentence (23)

Extraction gives rise to an unbounded dependency but in the case of pied-piping, a second unbounded dependency is introduced. Only the pronouns *qui*, *quoi* and *lequel* and the determiner *quel* allow pied-piping. Pied-piping means that the extracted constituent does not necessarily identify with the wh-word but it may concern a larger expression including this word.

In Sentence (23), the extracted constituent is *dans l' entreprise de qui* and the extraction is triggered by the relative pronoun *qui*. In this case, the relative pronoun represents a noun phrase more or less deeply embedded in the extracted constituent. Hence, there is a second unbounded dependency between the wh-word and the head of the extracted constituent.

Figure 20 shows the EPTD associated with the relative pronoun *qui* in the parsing of Sentence (23). The node *nNp* represents the noun phrase resulting from the modification of *Jean*, represented by the node *nNp0*, by the relative clause *dans l'entreprise de qui elle pense travailler bientôt*.

The extracted constituent *dans l'entreprise de qui* is represented by the node *nExtract*. This node shares the features `cat`, `prep`, and `ref` with *nTrace* but with different polarities.The node *nWh* represents a noun phrase reducing to the wh-word *qui* which may be embedded more or less deeply in the extracted constituent expressed with an underspecified dominance relation from *nExtract* to *nWh*. In Sentence (23), the chain of embedded constituents from *nWh* to *nExtract* contains a noun phrase (*qui*), a prepositional phrase (*de qui*), another noun phrase (*l'entreprise de qui*), and finally another prepositional phrase (*dans l'entreprise de qui*). Again, to restrict the possible chain to noun or prepositional phrases, a constraint is associated with the dominance relation from *nExtract* to *nWh*, in the form of the feature structure `{[cat:np|pp]}`.

A second underspecified dominance relation expresses that the trace of the extracted constituent may be embedded more or less deeply in a sequence of object clauses inside the relative clause. The most external object clause is represented by the node *nS1* and the most internal clause is represented by the node *nS0*. From the node *nS1* to the node *nS0*, there is an underspecified dominance relation. The following feature structure labels the underspecified dominance relation from *nS1* to *nS0*, to impose constraints on intermediate nodes: `{[cat:s|cs]`, `[funct:obj|obj_modal|obj_cpl|caus]}`. In our example, *nS1* and *nS0* are merged in the representation of the object clause *travailler bientôt*.

The phenomenon of extraction with pied-piping is not specific to French. For instance, it is also present in English, with the same difficulties. That is why the study of related work is not restricted to French. Analyses of this phenomenon were proposed within various formalisms. Categorial Grammar (CG) uses the paradigm of parsing as deduction (Moot and Retoré 2012): grammars are lexicalized and the elementary units attached at words are logical types; then, parsing amounts to a proof in a logical framework. The information necessary to realize extraction is attached at wh-words, in the same way as in IG.

Instead of an underspecified dominance relation in an EPTD, it is represented by a third order logical type. In the parsing process, which takes the form of a deduction, an additional hypothesis representing a trace of the extracted constituent is introduced and it will be discharged when the extracted constituent will be composed with the rest of the clause that follows. The standard logical kernel, the Lambek Calculus (Lambek 1958), is too rigid to express complex phenomena like middle extraction or islands to extraction. One solution is to enrich the logical framework with modal operators (Morrill 1994; Moortgat 1996). Another solution is to add new deduction rules to the restricted logical framework of AB-grammars (Steedman 2000).

HPSG uses feature structure unification to model the two unbounded dependencies present in extraction with pied-piping (Sag *et al.* 2003). Unlike IG and CG, the information is not only anchored at the wh-word: a non-local feature SLASH expresses that an argument or an adjunct of a word is lacking; a dual feature is introduced by the wh-word. Then, the two features are propagated up inside the constituency structure, and they meet at the top through a filler-gap mechanism.

In LFG, Kaplan and Zaenen (1995) use the mechanism of functional uncertainty to express long distance dependencies in constructions with extraction and pied-piping. In the c-structure, the rule representing the concatenation of the extracted constituent with the rest of the clause is associated with an equation expressing a sharing between two features in the f-structure: the first feature is attached at the extracted constituent and the second feature is represented by its path from the top to a deep level in the f-structure. The underspecification of this path, its uncertainty, is represented with a regular expression using the Kleene closure operator. The principle of functional uncertainty must be related to the underspecified dominance relations of IG and possibility of constraining them with feature structures, even if it is less general.

In TAG (Joshi and Schabes 1997), the adjunction operation makes it possible to represent the dependency of an extracted constituent from a distant predicative expression, but since it is more rigid than the mechanism of PTD superposition, it requires that information must be attached to the verb governing the clause at the source of the extraction. This contributes to the concentration of grammatical information

in verb entries. Moreover, taking pied-piping into account needs an ad hoc mechanism.

8 COMPARISON WITH OTHER FRENCH GRAMMARS AND EVALUATION OF THE GRAMMAR

There is very little work on the construction of French computational grammars from linguistic knowledge using semi-automatic tools. Historically, a very fruitful work was the PhD thesis of Candito (Candito 1999) about the modular organization of TAGs, with an application to French and Italian. This thesis was a source of inspiration for the development of several French grammars.

A first grammar produced according to this approach and able to parse large corpora was FRMG (Villemonte De La Clergerie 2010). FRMG falls within the TAG formalism and its originality lies in the use of specific operators on nodes to factorize trees: disjunction, guards, repetition, and shuffling. As a consequence, the grammar is very compact with only 207 trees. Moreover, these trees are not written by hand but they are automatically produced from a multiple inheritance hierarchy of classes (using a mechanism which is similar to the one used in XMG).

Another French grammar inspired by Candito (1999) is the French TAG grammar developed by Crabbé (2005). Like FRIGRAM, this grammar was written with XMG. Unlike FRMG, it is constituted of classical TAG elementary trees, hence its more extensive form: it includes 4,200 trees and essentially covers verbs. It was a purely syntactic grammar and it was later extended in the semantic dimension by Gardent and Parmentier (2007) for generation.

Evaluation of parsers is known to be a difficult task in general (Rimell and Clark 2008), mainly because each parser and each treebank is based on a large set of linguistic decisions ranging from the choice of category names and their definition to the choice of the head of constituents. Comparing a parsing result with a gold standard corpus requires conversion from one format to another and this conversion may induce biases.

In the case of FRIGRAM, to evaluate the accuracy and coverage is even more problematic for two reasons. First, when building

FRIGRAM, we decide to focus on a set of linguistic phenomena and cover them in an exhaustive way. At the same time, we also decide not to take into account (at least in this version of the grammar) several kinds of phenomena. We think that a fully lexicalized grammar is not the most efficient way of modeling some phenomena that break the basic structure of natural language sentences; for instance, dislocation, non-constituents coordination, parenthetical clauses, parenthesis, or direct reported speech are not taken into account. In Dufour-Lussier *et al.* (2014), we develop further this idea and explain how we plan to use an external and complementary tool, either as a preprocessing or postprocessing step to deal with these phenomena.

The second reason is that there is no robust parser able to deal with IG. The tool developed so far (LEOPAR; Guillaume *et al.* 2008) was designed to experiment with, test and help in grammar development. It was later enriched with filtering algorithms to improve the supertagging stages of the parsing process. Nevertheless, it does not have any robust mechanism to deal with sentences that are not completely covered by the grammar. After filtering steps, deep parsing relies on an exhaustive search of tree description models, which is an NP-hard task. As a consequence, LEOPAR can be used to parse sentences of length up to 15 words.

For the reasons given above, evaluation on treebanks is difficult and will need further experiments with a robust parsing algorithm and processing dedicated to phenomena not covered by the grammar. The French TreeBank (FTB; Abeillé *et al.* 2003) contains 12,351 sentences; 2,769 of these sentences have the length lower than 15. The parser LEOPAR with the FRIGRAM resource is able to parse 44.35% of the 2,769 sentences considered. More recently, a new French corpus, Sequoia (Candito and Seddah 2012), has been built and is freely available; it contains 3,099 sentences taken from different kinds of sources. Again, out of the 1,307 sentences of the length lower than 15, 52.8% are parsed. Unparsed sentences are mainly due to the lack of robustness or phenomena that FRIGRAM does not take into account (unusual coordination, incomplete negation or frozen expression for instance). In the case of the parsed sentences, due to the full search of solutions, the ambiguity can sometimes be high. In Figure 21, we can see that most of the parsed sentences have one or two solutions, but this number can grow up to 200 for some rare cases. In can be observed that

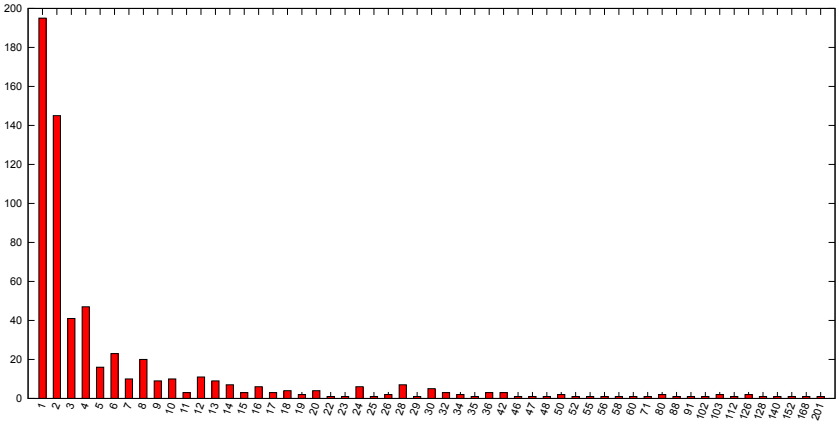


Figure 21: Frequency of sentences by number of solutions

most of the ambiguities are due to the PP-attachment; as we do not use semantic or statistical information so far, it is difficult to decide correctly for a PP-attachment in case of ambiguity.

To deal with this ambiguity, we have designed a set of rules that assign a weight to each dependency structure that is produced. It is then possible to estimate the accuracy of the grammar, comparing the best weighted parsing and the gold standard, by giving the labeled and the unlabeled attachment scores. Some preliminary work on ranking based on training on a corpus were conducted but it does not give a significant improvement with respect to hand-crafted rules. The labeled attachment score (LAS) is the proportion of dependency links that are correctly predicted by the grammar, whereas the unlabeled attachment score (UAS) is the proportion of dependency links that have the correct head disregarding the link label. We used Sequoia as the gold standard. On the successfully parsed sentences of Sequoia, the LAS is 81.6% and the UAS is 84.0%. Again, many errors are linked to PP-attachments not correctly ranked by our rules. Recently, partially supervised learning techniques were used to improve the performances of FRMG (Villemonte De La Clergerie 2013); with this hybridization technique, the LAS score of FRMG on Sequoia is 85.21%.

All results on freely available data can be found on the FRIGRAM web page. [12]

[12] `http://frig.loria.fr`

Raw corpora are useful to check for the robustness of a parsing system but they have some limitations concerning the coverage of the grammar. Test suites are built to overcome this limitation and give examples of a wider spectrum of grammatical phenomena. Such suites may include not only positive examples but also negative examples to test the overgeneration of the grammar. There exists such a suite for French, the TSNLP (Lehmann *et al.* 1996). On the set of grammatical sentences of the TSNLP, LEOPAR and FRIGRAM are able to parse 86% of the sentences. The remaining sentences correspond to sentences that should be covered by the robustness of the parser rather than by the detailed grammar (like, for instance, unusual kind of coordination, sentences with incomplete negations). For the ungrammatical sentences of the TSNLP, 37% are parsed by LEOPAR and FRIGRAM. The main sources of problems are: sentences that are syntactically correct but semantically incorrect, phonological rules, tricky rules for past participle agreement in French that are not encoded in the grammar.

To try to deal with TSNLP drawbacks, we have designed our own test suite which is complementary to the TSNLP; it contains 944 positive sentences and 192 negative ones. 97.5% of the grammatical sentences are parsed and the ratio is 19.8% for ungrammatical sentences. The reader can find the test suite and the parsing results on the same web page as before. When parsing succeeds, the list of dependency structures produced is also given. The variety of the examples gives a good idea of the coverage of FRIGRAM and the richness of dependency graphs helps to understand the subtlety of the grammar.

9 CONCLUSION

The next step to go ahead with FRIGRAM is to solve the bottleneck of the parser LEOPAR in order to parse raw corpora. We need to improve the efficiency of the parser to contain the possible explosion resulting from the increase of the grammar size in combination with the increased sentence length. It is also necessary to take robustness into account in the parsing algorithm and add extra-grammatical procedures to deal with phenomena that we do not want to model by the lexicalized grammar.

For English, Tabatabayi Seifi (2012) is the first attempt to build an interaction grammar, which should be extended in order to have a coverage equivalent to the one of FRIGRAM.

ACKNOWLEDGEMENTS

The authors wish to thank the reviewers for their useful remarks on previous versions of the article. They also thank Claire Gardent for her careful reading of the final version.

REFERENCES

Anne ABEILLÉ, Lionel CLÉMENT, and François TOUSSENEL (2003), Building a Treebank for French, in *Treebanks. Building and Using Parsed Corpora*, pp. 165–187, Kluwer Academic Publishers.

Anne ABEILLÉ and Danièle GODARD (1997), The Syntax of French Negative Adverbs, in *Negation and polarity: syntax and semantics*, pp. 1–27, John Benjamins publishing Company.

Anne ABEILLÉ, Danièle GODARD, and Philip MILLER (1997), Les causatives en français, un cas de compétition syntaxique, *Langue française*, 115:62–74.

Jason BALDRIDGE, Sudipta CHATTERJEE, Alexis PALMER, and Ben WING (2007), DotCCG and VisCCG: Wiki and Programming Paradigms for Improved Grammar Engineering with OpenCCG, in *Proceedings of the Grammar Engineering Across Frameworks Workshop (GEAF 07)*, pp. 5–25, CSLI Studies in Computational Linguistics ONLINE, Dagstuhl, Germany.

Emily BENDER (2008), Grammar Engineering for Linguistic Hypothesis Testing, in *Proceedings of the Texas Linguistics Society X Conference: Computational Linguistics for Less-Studied Languages*, pp. 16–36.

Guillaume BONFANTE, Bruno GUILLAUME, and Mathieu MOREY (2009), Dependency Constraints for Lexical Disambiguation, in *Proceedings of the 11th International Conference on Parsing Technology (IWPT 2009)*, pp. 242–253, Paris, France, `http://www.aclweb.org/anthology/W09-3840`.

Joan BRESNAN (2001), *Lexical-Functional Syntax*, Blackwell Publishers, Oxford.

Miriam BUTT, Helge DYVIK, Tracy H. KING, Hiroshi MASUICHI, and Christian ROHRER (2002), The Parallel Grammar Project, in *Proceedings of the Workshop on Grammar Engineering and Evaluation (COLING 2002)*, pp. 1–7, Association for Computational Linguistics, Stroudsburg, PA, USA.

Marie CANDITO and Djamé SEDDAH (2012), Le corpus Sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical, in

Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, pp. 321–334, ATALA/AFCP, Grenoble, France, `http://www.aclweb.org/anthology/F12-2024`.

Marie-Hélène CANDITO (1999), *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien*, Ph.D. thesis, Université Paris 7.

Jinho D. CHOI and Martha PALMER (2010), Robust Constituent-to-Dependency Conversion for Multiple Corpora in English, in *Proceedings of the 9th International Workshop on Treebanks and Linguistic Theories (TLT-9)*, pp. 55–66, Tartu, Estonia.

Ann COPESTAKE and Dan FLICKINGER (2000), An Open Source Grammar Development Environment and Broad-coverage English Grammar Using HPSG, in *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, pp. 591–600, Athens, Greece.

Benoit CRABBÉ (2005), *Représentation informatique de grammaires fortement lexicalisées : application à la grammaire d'arbres adjoints*, Ph.D. thesis, Université Nancy2.

Benoit CRABBÉ, Denys DUCHIER, Claire GARDENT, Joseph LE ROUX, and Yannick PARMENTIER (2013), XMG: eXtensible MetaGrammar, *Computational Linguistics*, 39(3):1–66.

Valmi DUFOUR-LUSSIER, Bruno GUILLAUME, and Guy PERRIER (2014), Parsing Coordination Extragrammatically, in Zygmunt VETULANI and Joseph MARIANI, editors, *Human Language Technology. Challenges for Computer Science and Linguistics. 5th Language and Technology Conference, LTC 2011, Poznan, Poland, November 25-27, 2011, Revised Selected Papers*, Human Language Technology Challenges for Computer Science and Linguistics, pp. 55–66, Springer International Publishing.

Catherine FUCHS, Nathalie FOURNIER, and Pierre LE GOFFIC (2008), Structures à subordonnée comparative en français. Problèmes de représentations syntaxiques et sémantiques, *Lingvisticæ Investigationes*, 31(1):11–61.

Claire GARDENT and Yannick PARMENTIER (2007), SemTAG: a platform for specifying Tree Adjoining Grammars and performing TAG-based Semantic Construction, in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pp. 13–16, Prague, Czech Republic.

Danièle GODARD (2004), French Negative Dependency, in Francis CORBLIN and Henriëtte DE SWART, editors, *Handbook of French Semantics*, pp. 351–390, Center for the Study of Language and Information.

Maurice GREVISSE and André GOOSSE (2008), *Le bon usage: et son édition Internet*, Grevisse langue française, De Boeck Supérieur.

Bruno GUILLAUME, Joseph LE ROUX, Jonathan MARCHAND, Guy PERRIER, Karën FORT, and Jenifer PLANUL (2008), A Toolchain for Grammarians, in *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008) - Demonstration*, pp. 9–12, Manchester, United Kingdom.

Bruno GUILLAUME and Guy PERRIER (2009), Interaction Grammars, *Research on Language and Computation*, 7:171–208.

Hélène HUOT (1982), Constructions infinitives du français : le subordonnant ”de”, *L'information grammaticale*, 15(1):40–45.

Angelina IVANOVA, Stephan OEPEN, Lilja ØVRELID, and Dan FLICKINGER (2012), Who Did What to Whom?: A Contrastive Study of Syntacto-semantic Dependencies, in *Proceedings of the Sixth Linguistic Annotation Workshop*, LAW VI '12, pp. 2–11, Association for Computational Linguistics, Jeju, Republic of Korea.

Aravind K. JOSHI, Leon S. LEVY, and Masako TAKAHASHI (1975), Tree Adjunct Grammars, *Journal of Computer and System Sciences*, 10(1):136–162.

Aravind K. JOSHI and Yves SCHABES (1997), Tree-Adjoining Grammars, in *Handbook of formal languages*, pp. 69–123, Springer.

Ronald M. KAPLAN and Annie ZAENEN (1995), Long-distance Dependencies, Constituent Structure, and Functional Uncertainty, *Formal Issues in Lexical-Functional Grammar*, 47:137–165.

Jong-Bok KIM and Ivan A. SAG (2002), Negation without Head-Movement, *Natural Language & Linguistic Theory*, 20(2):339–412.

Joachim LAMBEK (1958), The Mathematics of Sentence Structure, *Amer. Math. Monthly*, 65:154–170.

Sabine LEHMANN, Stephan OEPEN, Sylvie REGNIER-PROST, Klaus NETTER, Veronika LUX, Judith KLEIN, Kirsten FALKEDAL, Frederik FOUVRY, Dominique ESTIVAL, Eva DAUPHIN, Hervé COMPAGNION, Judith BAUR, Lorna BALKAN, and Doug ARNOLD (1996), TSNLP - Test Suites for Natural Language Processing, in *Proceedings of the 16th International Conference on Computational Linguistics (COLING 1996)*, pp. 711–716, Copenhagen, Denmark, `http://aclweb.org/anthology/C96-2120`.

Michael MOORTGAT (1996), Categorial Type Logics, in J. VAN BENTHEM and A. TER MEULEN, editors, *Handbook of Logic and Language*, pp. 93–177, Elsevier.

Richard MOOT and Christian RETORÉ (2012), *A Logic for Categorial Grammars: Lambek's Syntactic Calculus*, Springer.

Glyn V. MORRILL (1994), *Type Logical Grammar*, Kluwer Academic Publishers, Dordrecht and Hingham.

Stephan OEPEN, Dan FLICKINGER, Jun'ichi TSUJII, and Hans USZKOREIT, editors (2002), *Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing*, CSLI Lecture Notes, CSLI Publications, Stanford.

Guy PERRIER (2000), Interaction Grammars, in *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pp. 600–606, Sarrebrücken, Germany.

Guy PERRIER (2014), FRIGRAM: a French Interaction Grammar, Research Report RR-8323, Inria Nancy.

Carl J. POLLARD and Ivan A. SAG (1994), *Head-Driven Phrase Structure Grammar*, University of Chicago Press.

Geoffrey K. PULLUM and Barbara C. SCHOLZ (2001), On the Distinction between Model-Theoretic and Generative-Enumerative Syntactic Frameworks, in *Proceedings of the 4th International Conference on Logical Aspects of Computational Linguistics (LACL 2001)*, volume 2099 of *Lecture Notes in Computer Science*, pp. 17–43, Le Croisic, France.

Martin RIEGEL, Jean-Christophe PELLAT, and René RIOUL (1999), *Grammaire méthodique du français*, Presses universitaires de France.

Laura RIMELL and Stephen CLARK (2008), Constructing a Parser Evaluation Scheme, in *Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation (COLING 2008)*, pp. 44–50, Manchester, United Kingdom.

James ROGERS and K. VIJAY-SHANKER (1994), Obtaining Trees from their Descriptions: an Application to Tree-Adjoining Grammars, *Computational Intelligence*, 10(4):401–421.

Laurent ROMARY, Susanne SALMON-ALT, and Gil FRANCOPOULO (2004), Standards going concrete: from LMF to Morphalou, in Michael ZOCK, editor, *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pp. 22–28, Geneva, Switzerland, `http://acl.ldc.upenn.edu/W/W04/W04-2104.bib`.

Ivan A. SAG, Thomas WASOW, and Emily M. BENDER (2003), *Syntactic Theory: a Formal Introduction*, Center for the Study of Language and Information.

Mark STEEDMAN (2000), *The Syntactic Process*, Bradford Books, MIT Press.

Shohreh TABATABAYI SEIFI (2012), An Interaction Grammar for English Verbs, in Rasmus K. RENDSVIG and Sophia KATRENKO, editors, *Proceedings of the ESSLLI 2012 Student Session*, pp. 160–169, Opole, Poland, `http://ceur-ws.org/Vol-954/paper17.pdf`.

Karel VAN DEN EYNDE and Piet MERTENS (2003), La valence : l'approche pronominale et son application au lexique verbal, *Journal of French Language Studies*, 13:63–104.

Éric VILLEMONTE DE LA CLERGERIE (2010), Building factorized TAGs with meta-grammars, in *Proceedings of the 10th International Conference on Tree Adjoining Grammars and Related Formalisms (TAG+10)*, pp. 111–118, New Haven, CO, USA.

Éric VILLEMONTE DE LA CLERGERIE (2013), Improving a symbolic parser through partially supervised learning, in *The 13th International Conference on Parsing Technologies (IWPT 2013)*, pp. 54–62, Nara, Japan, https://hal.inria.fr/hal-00879358.

XTAG RESEARCH GROUP (2001), A Lexicalized Tree Adjoining Grammar for English, Technical Report IRCS-01-03, IRCS, University of Pennsylvania.

Nicholas YATES (2002), French Causatives: a Biclausal Account in LFG, in *Proceedings of the LFG02 Conference*, pp. 390–407, Athens, Greece.