Encoding a syntactic dictionary into a super granular unification grammar

Sylvain Kahane, François Lareau

Encoding a syntactic dictionary into a super granular unification grammar

Sylvain Kahane
MoDyCo, CNRS/Université Paris Ouest
sylvain@kahane.fr

François Lareau
OLST, Université de Montréal
francois.lareau@umontreal.ca

Abstract

We show how to turn a large-scale syntactic dictionary into a dependency-based unification grammar where each piece of lexical information calls a separate rule, yielding a super granular grammar. Subcategorization, raising and control verbs, auxiliaries and copula, passivization, and tough-movement are discussed. We focus on the semantics-syntax interface and offer a new perspective on syntactic structure.

1 Introduction

The encoding of large-scale syntactic dictionaries into formal grammars has been achieved many times since the 1990s. This paper presents the encoding of a large-scale syntactic dictionary in a dependency grammar (DG) characterized by extreme granularity. The first main contribution of this paper lies in the fact that each specification in the dictionary calls a separate rule. All rules are expressed in the same basic unification-based formalism in the form of elementary structures *à la* Tree Adjoining Grammar (TAG). The second contribution is that our syntactic dependency structure is richer than the usual representations in most DGs. It appears as a directed acyclic graph (DAG) from the point of view of the semantics-syntax interface, but as a proper tree for the syntax-text interface.

The formal framework in question, Polarized Unification Grammar (PUG), has been presented in various papers (Kahane, 2006; Kahane and Lareau, 2005; Lareau, 2008; Kahane, 2013), but the description of the lexicon in PUG has never been formally discussed. To see whether PUG could handle a wide range of lexico-syntactic phenomena, we built a formal lexicon-grammar interface on top of *Lexique des formes fléchies du français* (Le*fff*), a large-scale syntactic dictionary of French (Sagot, 2010). For the sake of clarity, we translated the entries discussed here into English and adapted some notations (without modifying the dictionary's architecture).

Unlike other unification-based grammars (Shieber, 1986; Francez and Wintner, 2011), PUG makes linguistic structure more apparent: we do not combine abstract feature structures, but geometrical structures such as graphs and trees. We have only one abstract mechanism, polarization, to control the combination of rules, so we do not need *ad hoc* features to do that. All the features in our structures correspond to lexical information that could not be suppressed in any framework. Thus, model artifacts are minimal.

Elementary PUG structures can combine to obtain less granular descriptions equivalent to TAG elementary trees. But unlike TAG, PUG uses the same mechanism for the combination of elementary pieces of information than for whole lexical units. In other words, it expresses both TAG's grammar and metagrammar (Candito, 1996; de La Clergerie, 2010) in the same formalism, which allows us to consider at the same time very fine-grained rules and rules with a larger span (routines, for instance).

In this paper, we focus on the semantics-syntax interface, including the mismatches between these two levels of representation, i.e., what generative grammar models in terms of movement. In our dependency-based approach, it is not words or constituents that are moved around, but rather the dependencies themselves, i.e., the relations between words (or constituents).

2 Lexical description

Le*fff*'s entries are divided into six fields, as illustrated below:

```
WANT; V; 'want'; ⟨1=subj:N|ProNom, 2=obj:N|ProAcc, 3=comp:to-Vinf⟩;
CtrlObjComp:to-Vinf; passive.
```

This entry contains the following information:

1. the lemma, WANT;
2. the part of speech V;[1]
3. the meaning, 'want';[2]
4. the subcategorization frame, giving for each semantic argument its syntactic realization; e.g., the third argument (3) realizes as an infinitive verb (Vinf) being a complement (comp) marked by TO;
5. control and raising features: CtrlObjComp:to-Vinf indicates that the object of WANT is controlled by (the "subject" of) the **comp:to** infinitive verb;
6. redistributions, such as passive voice for verbs or tough-movement for adjectives.

We slightly enriched the Le*fff* in three ways:

- We introduced an explicit numbering of semantic actants, useful for an interface with semantic description. It corresponds to the order in which actants are listed in the subcategorization frame.
- We slightly modified some entries for a better account of the distinction between control and raising, as well as tough-movement.
- We added entries for some lexical units that have a grammatical use, such as auxiliaries, that we encoded in Le*fff*'s format.

3 Model architecture

Our approach is based on Meaning-Text Theory (MTT), which views a linguistic model as a device that associates meanings to texts (Mel'čuk, 2016). Meanings are represented as graphs of predicate-argument relations between semantemes (i.e., signifieds of linguistic signs). We do not consider the phonological module here, so our texts are just strings of wordforms. Between meaning and form, we consider an intermediate level of representation, a syntactic dependency graph. Fig. 1 illustrates the full representation of sentence (1).

(1) *She slept.*

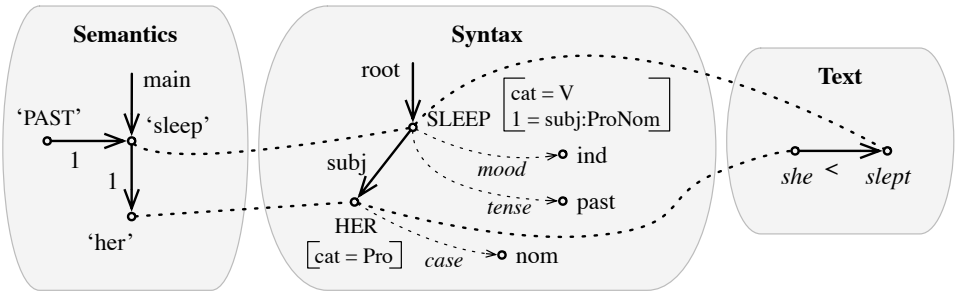

Figure 1: The three linguistic representations of (1)

The semantic representation contains two main kinds of objects: semantic nodes, labeled by semantemes (e.g., the lexical semanteme 'sleep' or the grammatical semanteme 'PAST') and semantic dependencies, labeled by a number r linking a predicate to its r-th argument. In addition to semantemes and predicate-argument relations, there is one pointer labeled "main" that flags one semanteme as the most salient; it corresponds roughly to the rheme's dominant node (Mel'čuk, 2001).

A syntactic representation comprises three kinds of objects: lexical nodes, labeled by lexical units (e.g., SLEEP), grammatical nodes, labeled with grammemes and linked to lexical units (e.g., "past" which

[1] Morphosyntactic subclasses, such as the conjugation group for verbs, are not considered here.

[2] Le*fff*'s entries do not contain this information; by default, we just recopy the lemma. We add it manually for grammatical words like $\text{BE}_{\text{copula}}$ and $\text{BE}_{\text{progressive}}$ (§5.2 and §5.3). Ideally, this field would distinguish senses and give a definition.

is a *tense*), and syntactic dependencies, labeled with syntactic functions (e.g., **subj**). All objects can bear features, such as "[cat=V]" on SLEEP.

A text representation contains two kinds of objects: nodes labeled by wordforms (e.g., *slept*), and linear precedence relations labeled "<".

Fig. 1 also shows another kind of object, correspondence links, represented by undirected dashed lines between nodes corresponding to the same linguistic sign across levels.[3] In the following sections, we will focus on the semantics-syntax interface.

4 The formal grammar

PUG generates a set of finite structures by combining elementary structures. A structure is a set of objects that are linked to three kinds of elements: 1) other objects (e.g., a dependency is linked to its source and target nodes), 2) atomic values (labels or feature values), or 3) polarities. All objects of elementary structures are polarized; this simple mechanism ensures that all necessary rules have been triggered without imposing any order on the combination of the rules (Kahane, 2006). The same rules are used for both analysis and synthesis, and the model allows incremental application strategies.

Polarities differ from atomic values in the way they combine. When two (elementary) structures combine, at least one object of a structure must be unified with an object of the other structure (as with TAG substitution, whereby the root of one tree is unified with a leaf of the other tree). When two objects are unified, all the elements linked to them must also be combined: objects and values are unified while polarities combine by a special operation called the product. We consider two polarity values in this paper: ▫ (white, unsaturated, or active), and ▪ (black, saturated, or inactive). Only ▫ can combine with other polarity values, and it is the identity element of the product: ▫ × ▫ = ▫; ▫ × ▪ = ▪; ▪ × ▪ = ⊥. Polarities should be interpreted as follows: white objects are unsaturated (they absolutely must combine with a non-white object and a final structure derived by the grammar must not contain any white object), while black objects are the elements of the structure constructed by the grammar. Objects are polarized by associating them to one of these two values via a function.

The grammar is modular: each module has its own polarizing function and also uses the polarities of adjacent modules to trigger their application (Kahane and Lareau, 2005). We consider here three levels (semantics, syntax and text) and two interfaces (semantics-syntax and syntax-text), giving us five modules. Instead of explicitly plotting the polarizing functions in our figures, we use five different geometric shapes, each associated with one module, as sketched in Fig. 2.

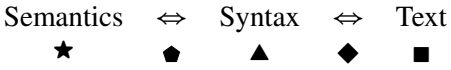

Figure 2: Modules of the grammar

We refer to modules by their proper polarizing function. For instance, $G^{\bigstar}$ is the semantic module, which builds semantic graphs, while $G^{\pentagon}$ is the semantics-syntax interface, which links semantic and syntactic objects. Each module is interfaced with its two adjacent modules (or only one for the modules at the ends of the pipeline). In consequence, a rule of a given module handles three polarities: the main polarity (the one proper to its module) and two articulation polarities (the ones of the adjacent modules). Generally, when an object's main polarity is saturated, its articulation polarities are white; they are used to trigger rules from adjacent modules, which are the only rules that saturate these polarities. We use black articulation polarities only when there are mismatches between two levels (e.g., with raising, when a semantic dependency does not correspond to a syntactic dependency linking the same lexical units). Indeed, an object with a black articulation polarity, being already saturated, does not trigger rules from the adjacent module. A rule always contains at least one object with a black main polarity. Objects with a white main polarity are used to specify the context and are not articulated.

Each object in a rule is typed and belongs to a specific level of representation. They all bear at least two polarities: ★ ⬠ for semantic objects, ⬠ ▲ ◇ for syntactic objects, and ◇ ■ for a surface object. Correspondence links, which will be introduced below, belong only to an interface and bear only the

[3] Dependencies also correspond pairwise, but links between them are not necessary for the implementation of this grammar.

interface polarity, ⬟ or ◆. To make our figures more legible, we only show the black polarities. But keep in mind that, for instance, a syntactic object drawn with only ▲ in a rule actually also bears ⬠ and ◇, since a syntactic object has always these (and only these) three polarities.

5 Encoding the lexicon

5.1 Lexicalization and government patterns

Let us start with the basic sentence (1) to show how modules are articulated. This first fragment of the grammar contains about twenty rules, which seems a lot for such a simple sentence. But most of these rules constitute the heart of the grammar and would be re-used for almost any sentence. The rules relating to specific lexical units' idiosyncrasies are directly derived from our lexical resource by instantiating a few very generic patterns. As we will show in the following sections, we do not need lots of additional rules to handle much more complex phenomena. For now, let us look at the following two lexical entries:

```
SLEEP; V; 'sleep'; ⟨1=subj:N|ProNom⟩; ∅; ∅.
HER; Pro; 'her'; ⟨∅⟩; ∅; ∅.
```

Our first module, G★ (Fig. 3), builds the semantic graph and calls G⬟. All objects constructed by these rules bear ★, while context objects bear ☆ (not shown). Remember that each ★ object is interfaced with G⬟ by bearing ⬠ as an articulation polarity (not plotted here). The rule $R^{\star}_{main}$ is the initial rule, marking one of the semantic nodes as the most communicatively salient meaning (which is normally realized as the syntactic root). $R^{\star}_{sleep}$ indicates that 'sleep' is a unary predicate; this is trivially derived from the subcategorization frame of SLEEP, which states that this lexeme has only one argument.

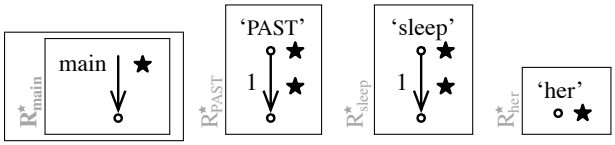

Figure 3: A fragment of G★

The next module, G⬟ (Fig. 4), ensures the lexicalization and arborization of the semantic representation.[4] The left part of the rule contains semantic objects (bearing ☆, not plotted here), while the right part contains syntactic objects (bearing △, also hidden). Semantic and syntactic objects that correspond to each other are linked by a correspondence link object. The objects constructed by a rule bear ⬟, while the others bear ⬠ (implicit). The two ⬠ correspondence links of $R^{\pentagon}_{1=subj:ProNom}$ (represented by dashed lines) ensure that the governors and dependents of the semantic and syntactic dependencies will be put in correspondence by a lexicalization rule that saturates them. $R^{\pentagon}_{main}$ indicates that the main verb can have the indicative mood (there could be competing rules to allow different roots). $R^{\pentagon}_{past}$ indicates that 'PAST' is realized by a grammeme. The dotted link labeled *tense* is a function linking the grammatical object to the lexical node and is not an object itself (and therefore is not polarized).

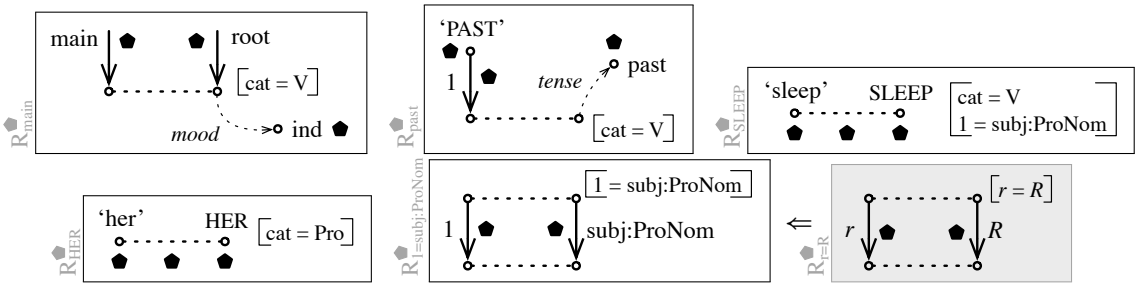

Figure 4: A fragment of G⬟

The lexical rules $R^{\pentagon}_{SLEEP}$ and $R^{\pentagon}_{HER}$ are directly induced by the dictionary: the meaning and lemma fields provide the labels for the semantic and syntactic nodes, the part of speech field provides the value for the "cat" attribute, and the subcategorization frame provides a list of features that are recopied as is

[4] We chose to present the grammar in the perspective of synthesis, from meaning to text, but this model is completely reversible and is compatible with various procedures, the cohesion of structures being completely ensured by polarization.

on the syntactic node. If present, control and redistribution features are also attached to the syntactic node and used as explained in §5.2. The actancial rule $R^{\pentagon}_{1=subj:ProNom}$ instantiates the generic pattern $R^{\pentagon}_{r=R}$ (grayed out). The syntactic dependency **subj:ProNom** is a "temporary" object and $R^{\pentagon}_{1=subj:ProNom}$ only makes sense when combined with $R^{\blacktriangle}_{subj:ProNom}$, presented below.

The module $G^{\blacktriangle}$ (Fig. 5) verifies the well-formedness of syntactic structures. The lexical rules of $G^{\blacktriangle}$ have been reduced to the minimum here, just to verify the general constraints related to parts of speech.[5] Grammatical rules verify that the grammemes *ind*, *past*, and *nom* appear in the right context. $R^{\blacktriangle}_{subj:ProNom}$ expresses the fact that **subj:ProNom** is indeed a subject dependency (**subj**), the dependent of which is a pronoun (*Pro*) with the nominative case (*Nom*). This is realized by "replacing" **subj:ProNom** with **subj**. In fact, there is no replacement, as both dependencies actually remain in the syntactic structure (both are syntactic objects), but only **subj:ProNom** is active for $G^{\pentagon}$ (it bears ⬠), while **subj** is active for $G^{\blacklozenge}$ (it bears ◇). This amounts to considering deep and surface functions (Fillmore, 1968; Blake, 2002). A dependency labeled by a surface function is validated by a rule such as $R^{\blacktriangle}_{subj}$, which sends it to $G^{\blacklozenge}$ just by leaving the articulation polarity ◇ white. Consequently, the syntactic dependencies built by $G^{\blacktriangle}$ form a DAG, from which only a subtree receives ◇ and thus is visible to $G^{\blacklozenge}$ and interfaced with the text.[6]

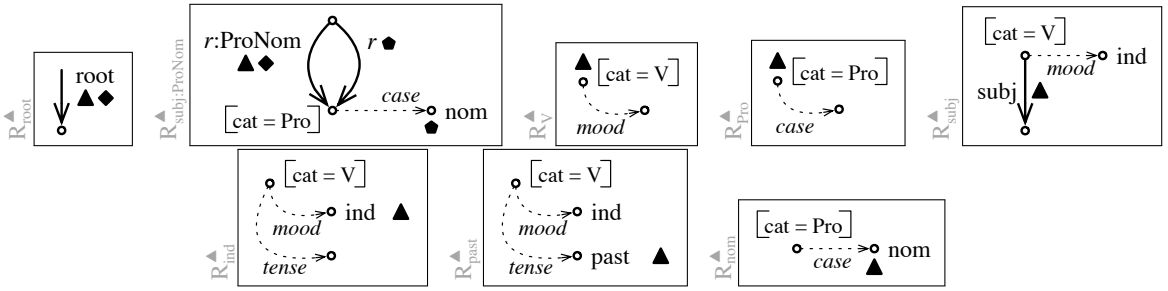

Figure 5: A fragment of $G^{\blacktriangle}$

The module $G^{\blacklozenge}$ (Fig. 6) ensures the linearization of the syntactic tree. $R^{\blacklozenge}_{subj}$ indicates that the subject can precede its governor without constraints. $R^{\blacklozenge}_{slept}$ indicates that the indicative past form of SLEEP is *slept* and $R^{\blacklozenge}_{she}$ indicates that the nominative form of HER is *she*. See (Kahane and Lareau, 2016) for a more detailed description of $G^{\blacklozenge}$, including rules for non-projective cases.

The module $G^{\blacksquare}$, which verifies that the output of $G^{\blacklozenge}$ is a string, is trivial and not discussed here.

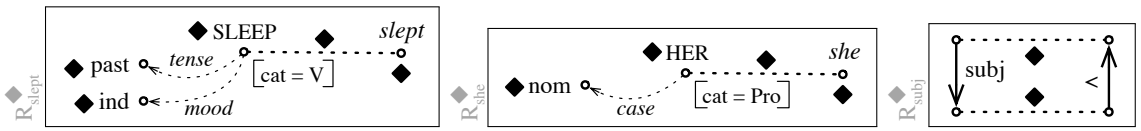

Figure 6: A fragment of $G^{\blacklozenge}$

5.2 Modification

Adjectives have at least one actant, which receives the **subj** fonction. However, it can never be realized as a subject on the adjective because a **subj** dependency can only be validated by $G^{\blacktriangle}$, which requires it to be headed by a verb (see $R^{\blacktriangle}_{subj}$, Fig. 5). In other words, the **subj** relation on the adjective must be "moved". Two solutions are possible: 1) the adjective is attributive and becomes a dependent of its first semantic actant ($R^{\blacktriangle}_{mod}$), or 2) the adjective is predicative and becomes the dependent of the copula ($R^{\blacktriangle}_{BE_{copula}}$), its subject becoming the subject of the copula ($R^{\blacktriangle}_{RaisSubjAux}$). The copula has no semantic contribution. Its lexical entry says that it has an adjective (Adj) or a past participle (Ved) as a dependent with the **aux** relation and that BE has the same semantic correspondent as its dependent. The rule $R^{\blacktriangle}_{RaisSubjAux}$ saturates a **subj** dependency (thus making it inert for $G^{\blacklozenge}$) and "replaces" it by a new **subj** relation (which receives ⬟ because it must not be interfaced with the semantic level). The triggering of $R^{\blacktriangle}_{RaisSubjAux}$ is only possible

[5] More specific constraints can be verified, such as the fact that a syntactic actant is obligatory, by introducing the corresponding syntactic dependency with △.

[6] It is very easy to write a PUG grammar to check that a structure is a proper tree (Kahane, 2006). We do not present this part of the syntactic module here.

if the verb has the feature "[RaisSubjAux]", which comes from the dictionary and is just recopied on the node. Its value is a boolean; by convention, its presence means it is "True", while its absence means "False". The same goes for the control and raising features in §5.3.

`RED; Adj; 'red'; ⟨1=subj:N|ProNom⟩; Ø; Ø.`

`BE`$_{\texttt{copula}}$`; V; Ø; ⟨0=aux:Adj|Ved⟩; RaisSubjAux; Ø.`

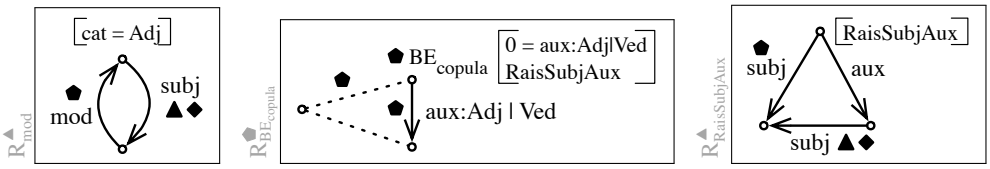

Figure 7: Adjectival modifiers

The rule $R_{BE_{copula}}$, as shown above, is in fact the combination of two simpler rules: a lexicalization rule that says BE_{copula} has no specific semantic contribution, and a rule that activates the "0=aux:Adj|Ved" instruction by mapping a single semanteme to a configuration of lexemes that jointly express that meaning (i.e., one is semantically empty):

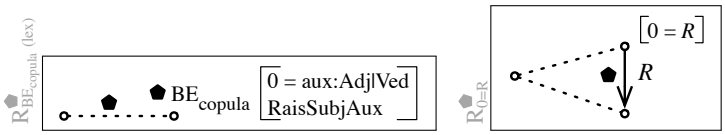

Figure 8: Decomposition of $R_{BE_{copula}}$

5.3 Control and raising

All verbs have a **subj** dependency, even infinitives and participles, but for these the **subj** dependency will not be active for $G^{\blacklozenge}$. This is achieved by rules of control and raising, which say that the **subj** dependency of the infinitive is realized on another verb. SEEM is a classical example of a raising verb; it has only one actant, which is realized as its complement (Fig. 9). Its subject can be an expletive ((2-b), $R^{\pentagon}_{IT_{expletive}}$) or the subject of its verbal dependent ((2-a), $R^{\blacktriangle}_{RaisSubjComp:to-Vinf}$).

`SEEM; V; 'seem'; ⟨1=comp:to-Vinf|that-Vind⟩; RaisSubjComp:to-Vinf; Ø.`

(2) a. *Ann seems to sleep.*
b. *It seems that Ann is sleeping.*

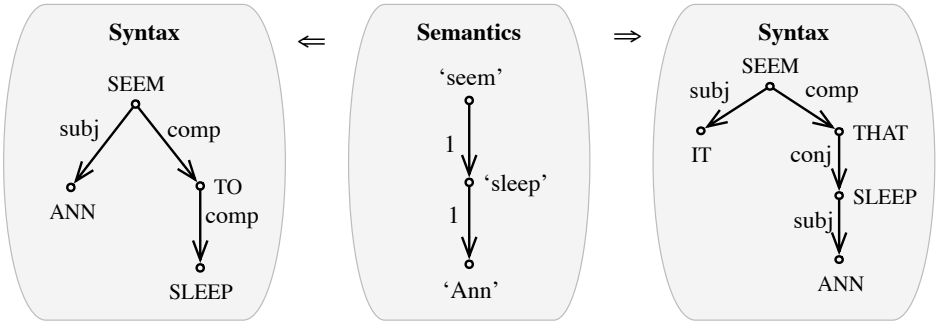

Figure 9: The simplified semantic and syntactic representations of (2).

A specification such as "[2=comp:to-Vinf]" will be expressed in three steps: a $G^{\pentagon}$ rule associates the semantic **2** dependency to a syntactic **comp:to-Vinf** dependency ($R^{\pentagon}_{2=R}$), which is first "replaced" by a **comp:to** dependency with a dependent that is a verb (V) with an infinitive mood (inf) ($R^{\blacktriangle}_{comp:to-Vinf}$), and then the **comp:to** dependency is "replaced" by a configuration with TO ($R^{\blacktriangle}_{comp:to}$). Even if as a result, **comp:to** is not active anymore for any of the interface modules $G^{\pentagon}$ and $G^{\blacklozenge}$, it is still part of the syntactic representation built by $G^{\blacktriangle}$ and it can be called as a needed context by the rules of raising and control.

Unlike SEEM, WANT controls its subject, which it shares with its verbal dependent. Therefore, contrary

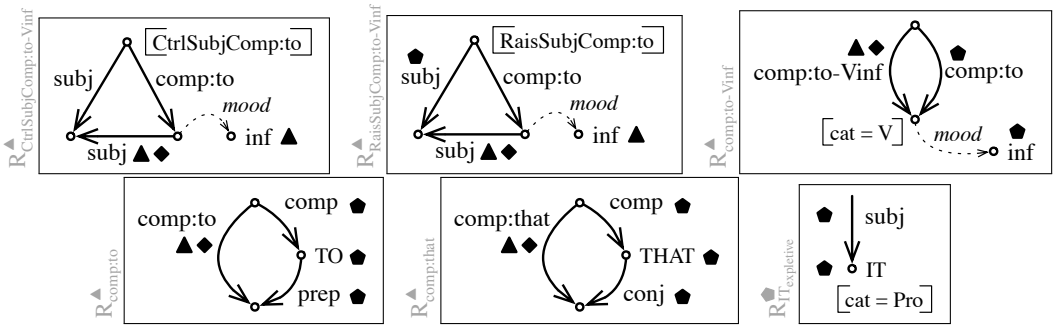

Figure 10: Rules for control and raising

to $R^{\blacktriangle}_{\text{RaisSubjComp:to-Vinf}}$, $R^{\blacktriangle}_{\text{CtrlSubjComp:to-Vinf}}$ does not block the semantic correspondence of the **subj** (cf. ⬟ on **subj** in $R^{\blacktriangle}_{\text{RaisSubjComp:to-Vinf}}$). Such rules are in fact $G^{\blacktriangle}$ rules validating the infinitive grammeme.

Tense-aspect-mood auxiliaries can also be described as raising verbs, such as the English progressive, which is a unary predicate expressed by the auxiliary BE imposing the gerundive form (Ving) on its unique actant and raising its subject:

$\text{BE}_{\text{progressive}}$; V; 'PROGRESSIVE'; ⟨1=aux:Ving⟩; RaisSubjAux:Ving; ∅.

5.4 Redistributions

In PUG, redistributions are dependency rewriting rules. For instance, the passive voice promotes the object to the subject position ($R^{\blacktriangle}_{\text{passiveObj}}$). The marker of this redistribution is the past participle, so such a rule can also be interpreted as a rule realizing this mood grammeme. As can be seen in Fig. 11, the **obj** dependency becomes inactive for $G^{\blacklozenge}$ and is replaced by a **subj** dependency inactive for $G^{⬟}$ but still active for $G^{\blacktriangle}$. The $R^{\blacktriangle}_{\text{passiveObj}}$ can be combined with another rule, $R^{\blacktriangle}_{\text{passiveSubj}}$, which demotes the subject to an agent complement (**comp:by**). The **obj** with a separate dependent ensures that the **subj** to be demoted is not an object that had been promoted by another rule.

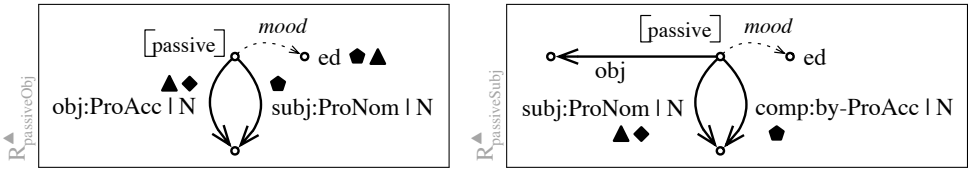

Figure 11: Passive

5.5 Tough-movement

Tough-movement is an interesting case of redistribution.

(3) a. *The book is easy to read.*
b. *a book easy to read*

We consider that in the expressions in (3) the initial subject of EASY has been demoted as a complement and that the object of its verbal dependent has been promoted in the **subj** position of EASY (Fig. 12). This is done by the rule $R^{\blacktriangle}_{\text{tough-mvt}}$, which combines a demotion rule and a raising rule (where an **obj** and not a **subj** is raised). The infinitive is not validated by $R^{\blacktriangle}_{\text{tough-mvt}}$, but by $R^{\blacktriangle}_{\text{inf-*ONE}}$, which suppresses the **subj** dependency but allows it to correspond to a general meaning 'one', as in *One reads the book.*

EASY; Adj; 'easy'; ⟨1=N|ProNom|Ving|to-Vinf|that-Vind⟩; ∅; tough-mvt.

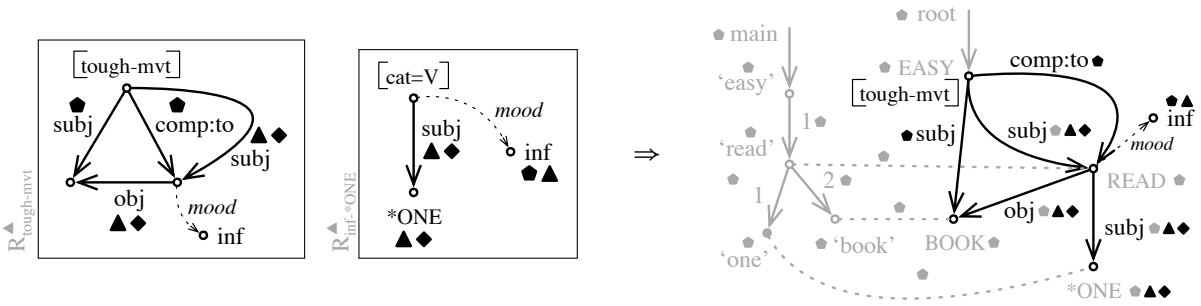

Figure 12: Tough-movement

6 Conclusion

The implementation of a dictionary as a formal grammar has been achieved in many frameworks (TAG, Lexical Functional Grammar (LFG), Head-driven Phrase Structure Grammar (HPSG), etc.). The dictionary we consider here has been used by TAG and LFG parsers for French (Boullier and Sagot, 2005; de La Clergerie, 2010). Implementing this dictionary in a new formal grammar is not challenging in itself. What should be interesting in our work is the fact that our grammar is very granular and each piece of information from the dictionary yields a separate rule. Moreover, the rules coming from the dictionary and the rules associated with grammatical items are quite similar in essence. It results from it that there is no real frontier between lexicon and grammar: what we obtain is a list of correspondence rules in the spirit of the *constructicon* of CxG (Goldberg, 1995; Fillmore et al., 2012), i.e., a dictionary describing both lexical units and more abstract constructions, such as dependencies, control and redistribution patterns, etc. That is, we have a grammar that describes linguistic signs, regardless of their lexical or grammatical nature. The cost of this granularity is that it is harder to implement, since PUGs must consider more possible object unifications than simple unification grammars. We are working on heuristics that will speed up unification in an implementation that is underway.

Another important point is that our grammar allows to understand some divergences in the literature concerning the nature of the syntactic structure. If we consider that the syntactic structure is the one containing all the objects with ▲, then our structure is richer than that of most frameworks; it contains various substructures considered by other formal grammars, *à la* (Hudson, 2007) (Fig. 13).

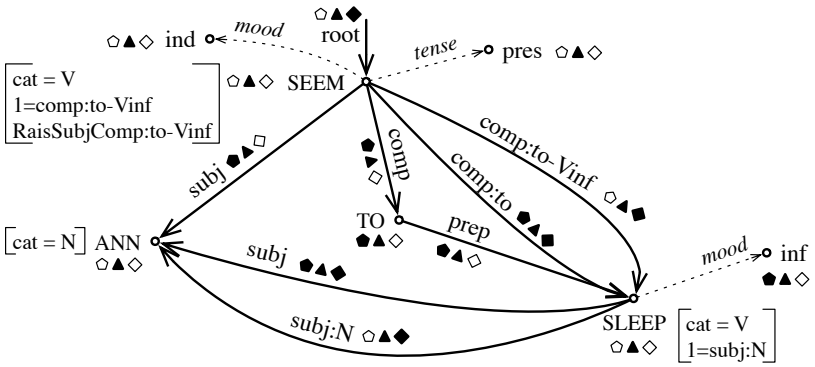

Figure 13: The whole syntactic structure of (2-a) produced by G▲

- The substructure that is interfaced with the text by G◆ (objects that bear ◇) is a surface syntactic tree, as considered by many dependency grammars. This structure contains both lexical nodes and grammatical nodes and has exactly one node per "syntaxeme" (lexeme or grammeme). Such a structure is more or less equivalent to a surface constituency tree in X-bar tradition, containing both lexical and functional projections.
- The substructure that is interfaced with the semantic graph by G⬟ (objects that bear ⬠) is a deep syntactic tree (Mel'čuk, 2012). This dependency structure contains only full words (e.g., TO is not part of this structure). It is a DAG, rather than a tree, some nodes having potentially two governors (in case of control, for instance).

- The whole structure can be compared to a phrase structure with movements (Graf, 2012; Stabler, 1997). A word can have several governors corresponding to several steps in the construction of the structure (cf. raising rules, for instance), which means that the same word occupies several positions alternatively. It is also comparable to the dependency structures considered by Word Grammar (WG) (Hudson, 2007) or Functional Generative Description (FGD) (Sgall et al., 1986).

Note that in PUG, "movements" do not duplicate the nodes in a structure. The number of nodes we consider never exceeds the number of syntaxemes. What we multiply are the dependencies between nodes, each encoding a syntactic combination of two items, which would be encoded in phrase structure trees by a new binary branching (Kahane and Mazziotta, 2015). In other words, we can involve a lexeme in various combinations without having to duplicate it, thus avoiding coindexation and movement.

References

Blake, B. (2002). *Relational grammar*. Routledge.

Boullier, P. and Sagot, B. (2005). Efficient and robust lfg parsing: Sxlfg. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 1–10. Association for Computational Linguistics.

Candito, M.-H. (1996). A principle-based hierarchical representation of ltags. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 194–199. Association for Computational Linguistics.

de La Clergerie, E. V. (2010). Building factorized tags with meta-grammars. In *The 10th International Conference on Tree Adjoining Grammars and Related Formalisms-TAG+ 10*, pages 111–118.

Fillmore, C. J. (1968). The case for case. In Bach, E. and Harms, R. T., editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart, and Winston.

Fillmore, C. J., Lee-Goldman, R., and Rhodes, R. (2012). The framenet constructicon. In Boas, H. C. and Sag, I. A., editors, *Sign-based construction grammar*, pages 309–372. CSLI Publications, Stanford.

Francez, N. and Wintner, S. (2011). *Unification grammars*. Cambridge University Press, Cambridge.

Goldberg, A. E. (1995). *Constructions: A construction grammar approach to argument structure*. University of Chicago Press.

Graf, T. (2012). Movement-generalized minimalist grammars. In Béchet, D. and Dikovsky, A. J., editors, *LACL 2012*, volume 7351 of *Lecture Notes in Computer Science*, pages 58–73.

Hudson, R. (2007). *Language networks: the new Word Grammar*. Oxford University Press.

Kahane, S. (2006). Polarized Unification Grammars. In *Proceedings of Coling-ACL 2006*, Sydney.

Kahane, S. (2013). Predicative adjunction in a modular dependency grammar. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 137–146, Prague. Charles University, Matfyzpress.

Kahane, S. and Lareau, F. (2005). Meaning-Text Unification Grammar: modularity and polarization. In *Proceedings of MTT 2005*, pages 163–173, Moscow.

Kahane, S. and Lareau, F. (2016). Word ordering as a graph rewriting process. In Foret, A., Morrill, G., Muskens, R., Osswald, R., and Pogodalla, S., editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Bozen, Italy, August 2016, Proceedings*, pages 216–239. Springer, Berlin/Heidelberg.

Kahane, S. and Mazziotta, N. (2015). Syntactic polygraphs. a formalism extending both constituency and dependency. In *Proceedings of the 14th Meeting on the Mathematics of Language (MoL)*, Chicago, USA. Association for Computational Linguistics.

Lareau, F. (2008). *Vers une grammaire d'unification Sens-Texte du français: le temps verbal dans l'interface sémantique-syntaxe*. Ph.d. thesis, Université de Montréal / Université de Paris 7.

Mel'čuk, I. A. (2001). *Communicative organization in natural language: the semantic-communicative structure of sentences*. Studies in language companion series. John Benjamins, Amsterdam/Philadelphia.

Mel'čuk, I. A. (2012). *Semantics: From Meaning to Text*, volume 1 of *Studies in Language Companion Series*. John Benjamins, Amsterdam/Philadelphia.

Mel'čuk, I. A. (2016). *Language: From Meaning to Text*. Ars Rossica, Moscow/Boston.

Sagot, B. (2010). The lefff, a freely available and large-coverage morphological and syntactic lexicon for french. In *7th international conference on Language Resources and Evaluation (LREC 2010)*.

Sgall, P., Hajičová, E., Panevová, J., and Mey, J. L. (1986). *The meaning of the sentence in its semantic and pragmatic aspects*. D. Reidel/Kluwer Academic, Dordrecht/Boston Hingham.

Shieber, S. M. (1986). *An Introduction to Unification-Based Approaches to Grammar*. CSLI Publications, Stanford, CA.

Stabler, E. (1997). Derivational minimalism. In *International Conference on Logical Aspects of Computational Linguistics*, pages 68–95. Springer.