Predicative Adjunction in a Modular Dependency Grammar

Sylvain Kahane

Modyco, Université Paris Ouest Nanterre & CNRS, France

`sylvain@kahane.fr`

Abstract

This paper shows how to introduce predicative adjunction in a dependency grammar inspired from TAG, MTT and RG. The addition of predicative adjunction allows us to obtain a modular surface syntactic grammar, the derivation structures of which can be interpreted as semantic representations.

1 Introduction

The dependency grammar we propose is inspired by TAG (Joshi 1987), MTT (Mel'čuk 1988) and Relational Grammar (Perlmutter 1980). Like TAG, we combine elementary tree structures in order to generate the surface syntactic structures of sentences. Also like TAG, we want our derivation structures to be interpretable as semantic structures (Rambow & Joshi 1994, Candito & Kahane 1998). Like MTT, our syntactic structures are dependency trees and our semantic structures are graphs of predicate-argument relations between the lexical and grammatical meanings of the sentence. Like RG, the syntactic structures are constructed by strata and a syntactic function can be revaluated. Such a formalism has actually already been fully established (see Nasr 1995, Kahane 2001, Kahane & Lareau 2005, and Lareau 2008). Kahane 2006 exlores the underlying formalism, which we call Polarized Unification Grammar (PUG), and shows that rewriting systems (including CFG), TAG, HPSG or LFG can be strongly simulated by PUG.

In this paper, we propose to write a PUG with rules (i.e. elementary structures) that are simple (from a mathematical point of view), but that cover rather complex linguistic phenomena, like extraction and complex determiners. The grammar makes extensive use of predicative adjunction, an operation borrowed from TAG for combining structures. Predicative adjunction adjoins the syntactic governor to a node, which means that the syntactic governor is the semantic modifier of its dependent (Shieber & Schabes 1994). As far as we know, the formalism we propose is the first genuine dependency grammar using predicative adjunction. With the help of predicative adjunction, elegant rules are possible that directly interpret the derivation structure as a semantic representation.

2 The base formalism

Polarized Unification Grammar (PUG) generates a set of finite structures by combining elementary structures. A structure is based on *objects*, for instance, on nodes and dependencies in a dependency tree. Objects are linked to three kinds of elements: 1) other objects (like a dependency to its source and target nodes), 2) *atomic values* (labels or feature values), and 3) *polarities*. Polarities differ from atomic values in the way they combine.

When two (elementary) structures combine, at least one object of a structure must be identified with an object of the other structure (like with TAG substitution, whereby the root of one tree is identified with a leaf of the other tree). When two objects are identified, all the elements linked to them must be combined: objects and values are identified (this is traditionally called unification in unification-based formalisms), while polarities combine by a special operation called the *product on polarities*. We consider three polarities in this paper:

> □ = white = unsaturated;
> ■ = black = saturated;
> ▪ = grey = invisible.

Only the white polarity can combine with other polarities and white is the identity element of the product:

·	□	■	▪
□	□	■	▪
■	■	⊥	⊥
▪	▪	⊥	⊥

Polarities can be interpreted as follows. White objects are unsaturated: they absolutely

must combine with a non-white object. A final structure derived by the grammar must not contain any white object. Black objects are the elements of the structure constructed by the grammar. Grey objects are introduced during the derivation but are "invisible" in the end.[1]

Fig. 1 proposes five rules for our dependency grammar. These rules are based on Nasr 1995, with improvements proposed by Kahane & Lareau 2005. Lexemes are represented by nodes and labeled with small capitals (SLEEP, BOY…). Syntactic dependencies are represented by downward pointing arrows. A third kind of objects, represented by diamonds, correspond to *grammemes*, that is, to what are more or less inflectional morphemes (Mel'čuk 1988).
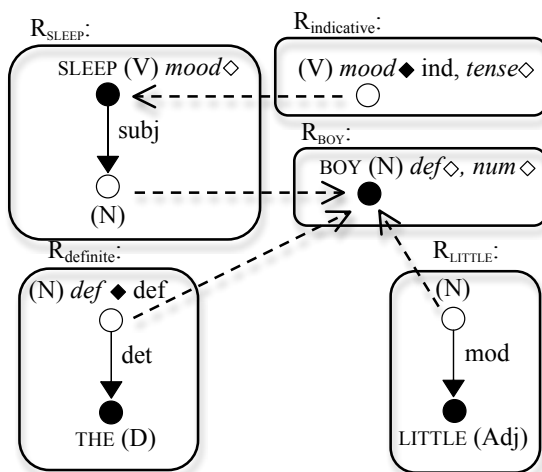

Figure 1. Sample rules and combination

The rule R_{SLEEP} indicates that the lexeme SLEEP needs a nominal subject and a mood (mood includes finiteness and its value can infinite as well as indicative). The subject dependency and the lexeme (polarized in black) are built by the rule, while the noun and the mood (polarized in white) are requests. The rule $R_{indicative}$ gives a verb the indicative mood

and says that the indicative must combine with tense.[2] The rule R_{BOY} introduces the noun BOY and asks for definiteness and number for it. Note that the request for a grammeme of definiteness forces the noun to take a determiner. The rule $R_{definite}$ adjoins THE to a noun and gives it a clear value for definiteness. The rule R_{LITTLE} adjoins LITTLE to a noun. These five rules can combine together, as suggested in Fig. 1 (cf. dashed arrows), yielding the dependency tree in Fig. 2. The result of such a derivation is called a *derived structure*.
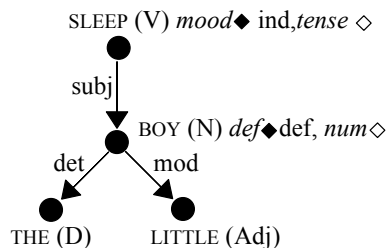

Figure 2. A (non-final) derived structure

Note that the tree in Fig. 2 is not saturated. It needs to combine with at least two more rules that introduce tense on SLEEP and number on BOY. We have also simplified the rule for the indicative mood: this mood not only asks for a tense, but for person and number agreement.
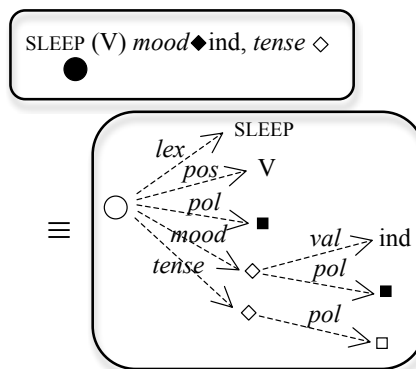

Figure 3. Our conventions[3]

Fig. 3 makes our formal conventions more precise. A black node has black polarity. The

---

polarity is the value of a map we call *pol*. The lexeme name and the part of speech (pos) are atomic values. Grammemes are objects linked to the lexeme node by a map labeled by the name of the inflectional category (mood, tense…). These objects have their own polarities and values in turn. When the rules R_{SLEEP} and $R_{indicative}$ are combined, two nodes are identified and the values of the maps *pol, pos* and *mood* common to both of them are unified. The values of the map *mood* and *tense* are objects, so they must be identified and their own maps unified and so on.[4]

The structure showing the way of how the rules combine in a derivation is called a *derivation structure* (see Vijay-Shanker 1992 for TAG). We consider three ways of combining rules. The first way is by *substitution*: R_{BOY} substitutes in R_{SLEEP} because R_{BOY} saturates a leaf of R_{SLEEP}'s structure. The second way is by *adjunction*: R_{LITTLE} adjoins to R_{BOY} because R_{LITTLE} adds a dependency to a black node of R_{BOY}'s structure.[5] The third way is *grammatical completion*: $R_{definite}$ completes R_{BOY} by saturating a grammeme in R_{BOY}'s structure.
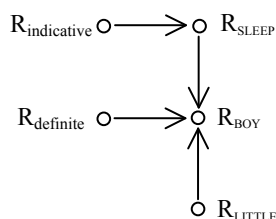

Figure 4. Derivation structure

Fig. 4 shows the derivation structure associated with the derivation suggested in Fig.1 giving the derived structure in Fig. 2. We adopt the following conventions: a substitution is represented by a downward arrow, an adjunction by an upward arrow, and an inflectional completion by a horizontal arrow. With these conventions, the derivation structure can

be interpreted as a graph of predicate-argument relations (as shown in Candito & Kahane 1998 for TAG) and is the basis for a semantic representation (see Mel'čuk 2012). Indeed, each arrow in the derivation structure can be interpreted as a semantic dependency that points from a predicate to one of its arguments.

To conclude this section, it must be remarked that we focus on the syntax-semantics interface in this paper and we do not discuss word order. Gerdes & Kahane 2001 propose a formalism—*topological grammar*—for linearizing dependency trees, even for languages with non-projective constructions. It is possible to write a topological grammar in PUG and to combine it with the grammar presented here (see Kahane & Lareau 2005 for the combination of different modules in PUG).

3   Dependency rules

The first improvement to the base grammar that we propose is to separate the rules associated with the lexemes proper from the *dependency rules* ensuring the realization of a given dependency (see the nodal vs. sagittal rules of Kahane & Mel'čuk 1999). We modify our previous rules as a consequence and add separate rules for dependencies. See Fig. 5, which shows two rules the combination of which ($L_{SLEEP} \oplus D_{subject}$) results in R_{SLEEP}.[6]
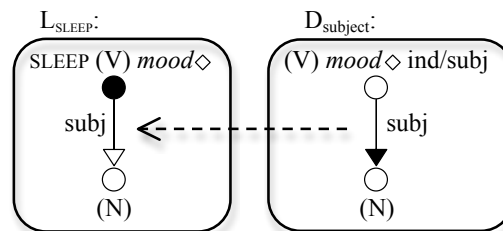

Figure 5. Lexical vs. dependency rules[7]

The dependency rule $D_{subject}$ constrains the subject dependency to be saturated only if the verb has the indicative or subjunctive mood. If

---

[4] We have slightly simplified Fig. 3. A map *host* from each grammeme to its lexeme is needed in order to ensure that the identification of two grammemes entails the identification of their host lexemes.

[5] This kind of adjunction is called *sister adjunction* by Rambow et al. 1995. Contrary to predicative adjunction, sister adjunction does not change the weak generative capacity of the formalism. One can note that the distinction between substitution and adjunction is small in PUG: in both cases, a black and a white node are identified. The only difference is the direction of the dependency, which is not really relevant in PUG (source and target are interchangeable from a graph-theoretical perspective).

[6] We will use the following conventions to name our rules: L for rules instantiating a lexeme, D for a dependency, G for a grammeme and S for structural rules that do not instantiate any object. R is an umbrella identifier used in Section 2, when lexical rules and dependency rules were not separated.

[7] In fact, a black dependency has to combine with a white one (if not, a second subject relation could be added to a verb). This can be easily solved by adding a second polarity: every black dependency will have a white additional polarity, while a white dependency will have a black one (see also positive and negative polarities in Kahane 2006).

the subject requirement is lexical (the subject is the first actant of the verb, cf. Tesnière 1959), the subject realization is controlled by the mood (mood includes finiteness) (cf. the position of the subject under IP in X-bar syntax). When the verb has another mood, like infinitive or participle, the subject requirement of L_{SLEEP} is not realized by a dependency on SLEEP. For instance, with a progressive form (*is sleeping*), the subject is lifted to the auxiliary BE. The subject requirement of SLEEP unifies with a grey polarity (and becomes invisible for other modules) and is replaced by a subject dependency on BE. Fig. 6 shows the rule ($G_{progressive}$) and the result of its combination with L_{SLEEP} ($L_{SLEEP} \oplus G_{progressive}$).[8]
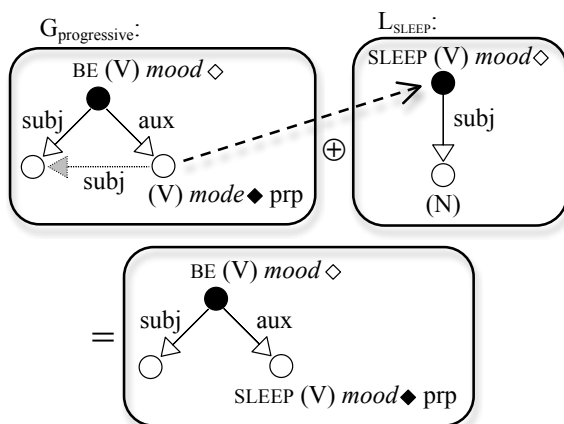
Figure 6. Invisible dependency.[9]

The fact that our formalism allows us to suppress a dependency and to replace it with another one is similar to what is proposed by RG. This leads us to consider deep and surface functions (Fillmore 1968). A *surface function* appears in a surface syntactic tree. A surface function must have been instantiated by a saturation rule, like $D_{subject}$ for the subject function (Fig. 5). A *deep function* is the initial function received by a lexeme, which can have been instantiated into an identical surface function or which may have been made invisible and replaced by another function.

Some deep functions will never be instantiated as surface functions. This is the case with

[8] A similar rule has been proposed by Kahane (2001), where a grey dependency is called a quasi-dependency. The idea to consider a quasi-dependency as a dependency with a saturated interface polarity occured later (Kahane & Lareau 2005) and is explained in Lareau 2008.

[9] We do not represent grey objects in derived structures in order to simplify the figures and to emphasize the fact they are invisible for further treatments.

the *to-obj* of TALK, which must be marked by the preposition TO. Fig. 7 shows the rule for the lexeme TALK and the *to-obj* dependency, as well as their combination ($L_{TALK} \oplus D_{to\text{-}obj}$). The last schema in Fig. 7 is a part of the derivation structure of the sentence *Aya talks to Bob*; it shows that L_{BOB} combines directly with L_{TALK}, while $D_{to\text{-}obj}$ is not interpreted as a lexical rule (TO is only a syntactic marker), but as a revaluation of the connection between two lexical rules. Such a rule is ignored when the derivation is interpreted at the semantic level.
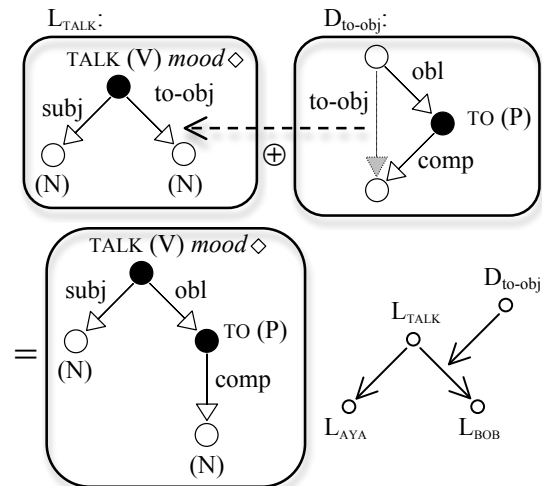
Figure 7. Governed preposition.

We choose not to introduce TO in the elementary structure of TALK contrary to what is done in TAG or in previous versions of our dependency grammar (Kahane 2001). We want our grammar to be as modular as possible, separating everything that can be separated. The government markers must be separated from the lexemes that call them for at least two reasons:

- they can be repeated in coordination: *Aya talks **to** Bob and **to** Dave*.

- they can have an alternative realization; for instance, in French, an indirect object is marked by the preposition À for a noun phrase, but by the dative case for a clitic pronoun (Fig. 8).[10] In English, a complementizer can be realized or not: *Aya knew* (*that*) *Bob came*.

[10] The two constructions of PARLER 'talk' are:
  (i) *Aya parle **à Bob***   'Aya talks to Bob'
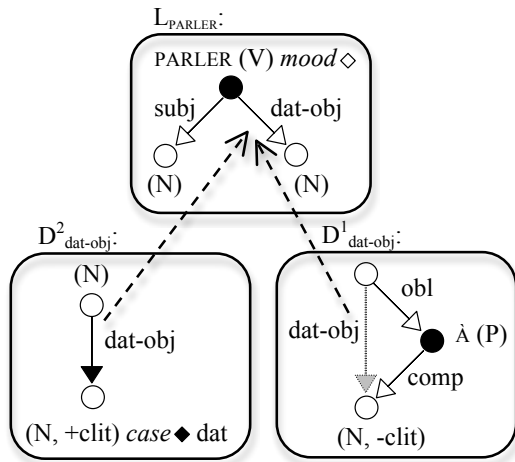 (ii) *Aya **lui** parle*   'Aya talks to him'

Figure 8. Indirect object in French

Therefore, we want the government markers to be realized by separate rules, but, for semantic reasons, we also want a predicative lexeme to combine directly with its arguments. We have solved these constraints thanks to the grey polarity and the invisible objects; as is shown in Fig. 7, the verb TALK has a direct connection with its semantic argument and it is only in the derived structure that the preposition appears between the verb and its argument.

The case of adjectives is interesting. Adjectives occur in two basic constructions. In the attributive construction they modify a noun (*the red book*) and in the predicative construction they form a verbal complex with the copula (*the book is red*). In both cases, 'red' is a semantic predicate and 'book' is its semantic argument. But some adjectives take an event as argument and this argument can be realized by a verb (***reading this book** is tough*). Such adjectives cannot modify their verbal argument, but the direct object of this argument can be promoted (so-called *tough-movement*) and become the syntactic governor of the adjective (***a book** tough to read*). According to these facts, we cannot assume that the attributive construction is the base construction of every adjective, since this construction is impossible for adjectives taking a verbal argument. We thus propose that, in its base construction, the argument of the adjective has a special deep function we call subj' (Fig. 9). This deep function does not exist as a surface function, but at least three constructions can apply to it: the attributive construction ($D_{subj'}$), the predicative construction (S_{copula}) and the construction of tough-movement ($S_{tough-mvt}$). Note that the at-

tributive construction can only apply if the argument is realized as a noun or after tough-movement, which promotes a noun as subj'.

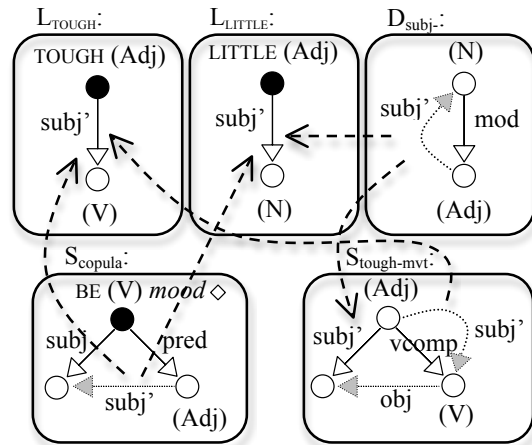
Figure 9. Adjectival constructions

The same deep function subj' is used for the passive past participle, which can enter into a predicative construction (*a book **has been stolen***) or an attributive construction (*the book **stolen by Bob***). The passive construction marked by the past participle inflection can be derived from a verbal base form by a rule using grey polarities again (Fig.10).
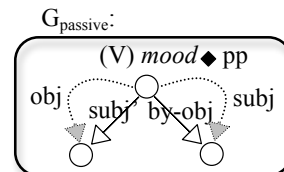

Figure 10. Passive voice

Intermediate conclusion: Thanks to the grey polarities, which allow us to make an object invisible (for other modules) and to replace it with another configuration, we are able to separate the rules describing the various constructions associated with a lexeme from the rule describing the lexeme itself. In a lexicalized grammar like TAG, an elementary structure describes a lexeme with one of its constructions and the set of elementary structures associated to a given lexeme must be generated by another module (like a metagrammar in Candito 1996). An advantage of the approach presented here is that the rules associated with the lexemes and their constructions are in the same formalism. They can be precompiled to obtain a lexicalized grammar or they can be triggered on-line during text analysis or synthesis.

141

We will now see how grey polarities can be used for modeling another way of combining lexemes: predicative adjunction.

4 Complex determiners

Numerous determiners are idiomatic constructions like *loads of* in (1):

(1) *Aya read **loads of** books*.

Such a construction causes a mismatch between the syntactic and the semantic structures: *loads* is the syntactic head of the NP *loads of books* and thus the syntactic dependent of *read*, but 'book' is its semantic head and the argument of 'read'. As a consequence, we want the derived tree and the derivation structure of (1) to be as in Fig. 11 (besides $S_{insertion}$, which is a technical rule introduced below).

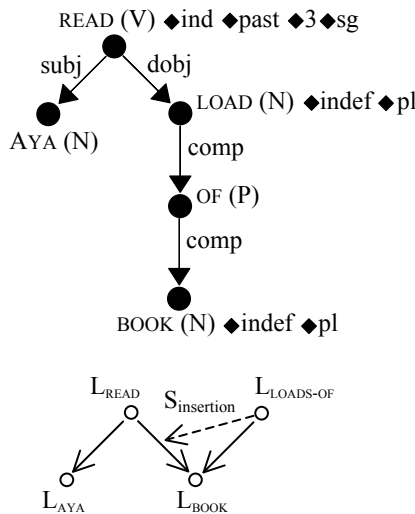READ (V) ◆ind ◆past ◆3◆sg

subj   dobj

AYA (N)   LOAD (N) ◆indef ◆pl

comp

OF (P)

comp

BOOK (N) ◆indef ◆pl

L_{READ} $S_{insertion}$ $L_{LOADS-OF}$

L_{AYA} L_{BOOK}

Figure 11. Derived tree
and derivation structure for (1)[11]

This problem can be solved by *predicative adjunction*, as in the standard analysis of complex determiners in TAG (Shieber & Schabes 1994). A predicative adjunction is an adjunction where the adjunct is inserted in the syntactic governor position. This is making possible in our formalism by way of a special rule, $S_{insertion}$, allowing the insertion of a node and a dependency between two other nodes (Fig. 12). This rule can be compared to a rule like D_{to-obj}, which introduces a marker. But $S_{insertion}$ is a generic rule that can apply on any dependency (r is a variable) and does not replace it but only inserts a dependency [*insert*:+] be-

[11] We have simplified the derived tree by suppressing the attribute before the grammemes.

hind. The formalism forces us to replace the r dependency by a new one, but in fact we just want to "move" it to a new dependent.
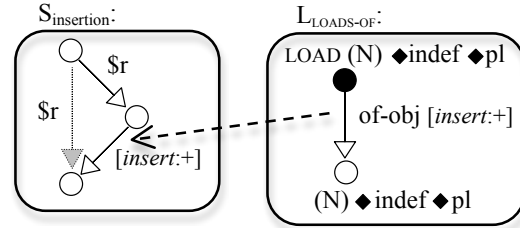

Figure 12. Predicative adjunction[12]

The predicative adjunction is indicated by a downward arrow in the derivation structure, like a substitution (Fig.11). A purely structural rule like $S_{insertion}$, which does not saturate any object, is represented by a dashed arrow. Note that L_{BOOK} substitutes in $L_{LOADS-OF}$ like in a normal substitution. Consequently, L_{BOOK} substitutes in two rules (L_{READ} and $L_{LOADS-OF}$), which is made possible by $S_{insertion}$. It is important to note that it is BOOK and not LOAD that fulfill the object position of READ. Therefore determiners that (predicatively) adjoin do not need to be nouns. In French there is a productive construction where the complex determiner is an adverb governing the noun (*trop de* N 'too_much of N', *moins de N* 'less of N' …). It is also possible to treat simple determiners, including articles, as syntactic governors of the noun they determine (Abney 1987, Hudson 2007). But even if in *read a book* the determiner A becomes the governor of BOOK, it is the noun of the unit *a book* that will combine with READ and it is not really legitimate to consider *a book* as a DP.

Note that the rule $S_{insertion}$ can be applied recursively, as in **more than half of all** *known species are insects*; indeed *half of* predicatively adjoins to *all known species* and *more than* predicatively adjoins to *half*.

5 Extraction

We will focus on relative clauses and start with *wh-relative clauses*, that is, with relative clauses marked by a wh-word, like in (2):

(2) (*the guy*) **who** *I invited*.

A relative clause depends on a noun (here *guy*), which is the antecedent of a wh-word (here *who*). Each wh-word will have its own

[12] Again the preposition (*of* in *loads of books*) is added by a separated rule.

rule: the rule S_{WHO} attaches the wh-word WHO to a noun bearing the feature +human (Fig. 13). The rule D_{wh-rel} allows predicative adjunction on a wh-word.[13] The rule D_{rel} instantiates the dependency between the antecedent noun and the relative clause and forces the head of the relative clause to be a finite verb in the indicative or subjunctive mood (Fig. 14).
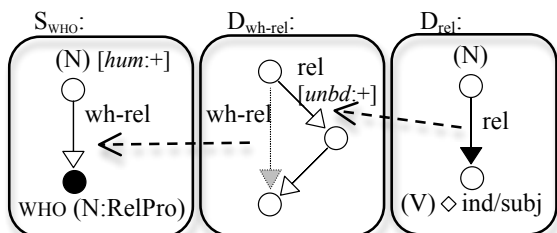

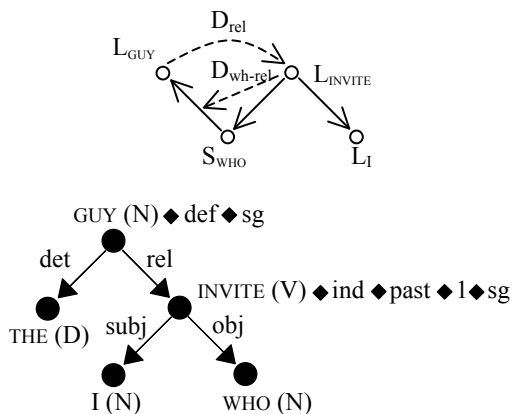
Figure 13. Rules for wh-relatives

Figure 14. Derivation structure
and derived tree for (2)

Tesnière 1959 argues that the wh-word occupies two positions in the dependency structure: it is both a complementizer (thus being the syntactic head of the relative clause) and a pronoun (filling the "extracted" position) (see additional arguments in Kahane 2002). Even though the wh-word occupies only the pronominal position in the derived tree in our account, the wh-word directly attaches to the antecedent noun (see S_{WHO} in Fig. 13 and in the derivation structures of Fig. 14 and 16) and thus carries out a complementizer role. It is even possible to not make invisible the wh-rel dependency between the antecedent noun and the wh-word and to obtain a dependency structure similar to Tesnière's stemma for extraction.

---

[13] We could have proposed a rule directly combining S_{WHO} and D_{wh-rel}, as it was done in previous works (Kahane 2001).

It has been well established since Ross 1967 that the string of dependencies between the antecedent noun and the "extracted" position (that is, the position filled by the wh-word) is potentially unbounded but nevertheless very constrained. In (3), the verb THINK has been added in the relative clause and has become the syntactic head of the clause.

(3) (*the guy*) *who **you think** I invited*.

Such a verb is called a *bridge verb*. We introduce a new rule, $S_{unbounded}$, allowing the predicative adjunction of a bridge verb (Fig. 15). Note that only syntactic dependencies labeled [*bridge*:+] can be inserted behind a dependency labeled [*unbd*:+]. This is comparable to the LFG's functional uncertainty (Kaplan & Zaenen 1989), where the constraints on extraction are also controlled at the syntactic function level.
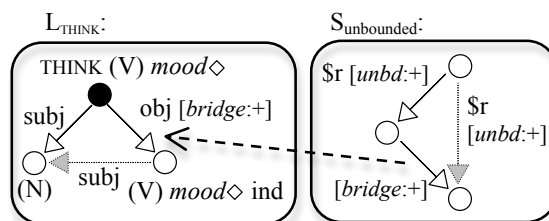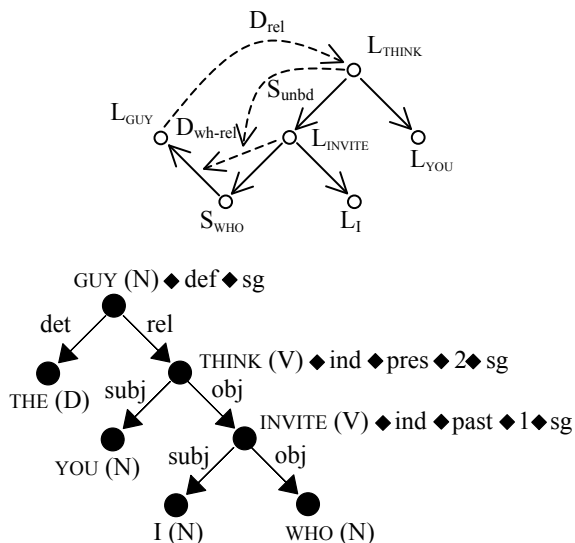

Figure 15. Rules for bridge verbs

Figure 16. Derivation structure
and derived tree for (3)

In the derivation of (3), who adjoins on GUY and substitutes into the object position of INVITE (Fig. 16). The latter is then inserted between WHO and GUY thanks to S_{wh-rel}, which introduces a rel(ative) dependency. In the same

143

way, INVITE substitutes in the object position of THINK. As the object dependency bears a feature [*bridge*:+], the rule $S_{unbounded}$ can apply and insert THINK between GUY and INVITE, replacing the rel relation by a new one. This can be done recursively. At the end, the rel dependency is instantiated by D_{rel}.

Our analysis of wh-relative clauses is similar to the TAG analysis, where the bridge verbs predicatively adjoin (Kroch & Joshi 1986). It can also be compared to the analysis in other frameworks, in particular to LFG with the functional uncertainty, to the generative grammar with Move α, or to HPSG with the slash feature. Contrary to the Move α or slash analyses, it is not the wh-word that is moved, but the bridge verbs that are inserted, like in the functional uncertainty. Moreover, it is not the verb THINK that is a bridge, but only its object dependency, like in LFG again (Kaplan & Zaenen 1989). Nevertheless, $S_{unbounded}$ is a rule that moves the rel dependency from INVITE to THINK, which can be seen as movement of the complementizer role of the wh-word.

We will now contrast wh-relatives with *that-relative clauses*, like (4):

(4) (*the guy*) ***that*** *you think I invited*.

It has often been argued that that-relatives are syntactically different from wh-relatives, because 1) THAT also marks complement clauses (*you think **that** I invited this guy*), 2) it does not accept pied-piping (*the guy to whom I speak* vs. **the guy to that I speak*), 3) that-clauses alternate with unmarked clauses (*the guy you think I invited*), and 4), contrarily to wh-words, THAT does not have a "human" feature. Consequently, THAT is not analyzed as a pronoun like wh-words, but only as a complementizer, which marks the subordination of a finite clause. We thus necessitate different extraction rules for that-relatives, where the antecedent noun directly fills the "extracted" position: $D_{obj-extraction}$ and $D_{subj-extraction}$ (Fig. 17). Subject and object extraction are differentiated because 1) only the object extraction really accepts the predicative adjunction of bridge verbs (?**the guy that you think that invited me*), and 2) only the object extraction can be unmarked (*He is the guy *(that) invited me*). Consequently, only $D_{obj-extraction}$ can combine with $S_{unbounded}$ and $D_{unmarked-relative}$.

With this new set of rules we obtain a different syntactic structure (Fig. 18): THAT is the syntactic head of the relative clause and contrarily to WHO, THAT does not fill the "extracted" position and does not combine with INVITE in (4). THAT only intervenes to mark the subordination of the relative clause.
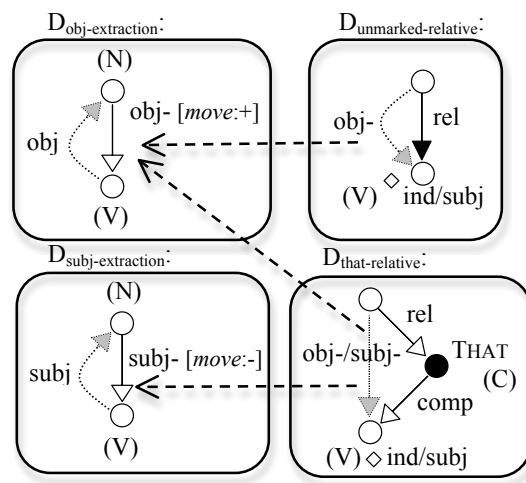

Figure 17. Rules for that-relatives
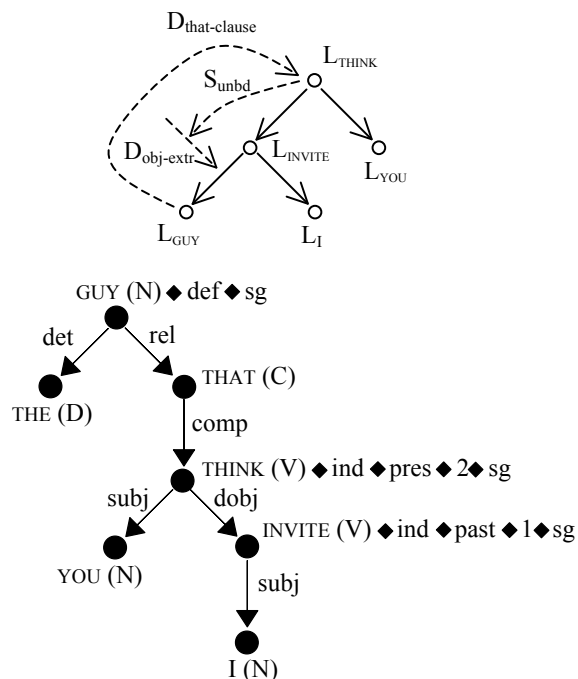

Figure 18. Derivation structure and derived tree for (4)

The case of French confirms the relevance of the distinction between wh-relatives and that-relatives. French has a +human wh-word QUI, comparable to WHO, but which can only be used after a preposition (*la fille à **qui** je parle* 'the girl to whom I talk'). In case of object extraction, the relative clause is marked by

QUE, which can be compared to THAT (*le livre que je lisais* 'the book that I read'); in particular, QUE is also the marker of complement clauses (*tu penses que j'ai invité ce gars* 'you think that I invited this guy') and no pied-piping is possible with QUE. Subject extraction is marked by a QUI form, different from the other QUI, because this one is not +human (*le livre qui est là* 'the book that is there'). Kayne 1975 argues that QUE is always a pure complementizer, even when it marks relative clauses. His main argument is the so-called *qui-que* alternation:

(5) (*la fille*) **que** *je pense* **qui** *m'a invité*
    the girl that I think   that invited me
    'the girl that I think invited me'

In (5), the subject of the complement clause is extracted, but, contrarily to what might have been expected, the marker of the relative clause is QUE (*la fille que* …) and moreover the marker of the complement clause is QUI (*je pense qui* …). As a consequence, we can conclude that QUE and QUI are not exactly the subject and object relative markers and that they are not selected according to the "extracted" position they fill. We propose rather that these markers are selected according to the verb they complementize: QUI complementizes a verb without a realized subject, while QUE must complementize a verb with a subject. Such a solution can only be implemented in our formalism if QUI and QUE are treated as markers and connected to the verb they complementize in the syntactic tree (which is not the case for WHO in our analysis). Consequently, we propose subject and object extraction rules for French similar to the rules for English (of Fig. 17), but we add a ±subject feature on the subordinated verb indicating whether its subject has been extracted or not ($D_{subj-extraction}$ in Fig. 19). The QUI marker can only mark a relative clause if the main verb of the clause is [*subj*:-] ($D_{qui-relative}$ in Fig. 19).

The rule for relatives marked by QUE is similar ($D_{que-relative}$ in Fig. 20); the only difference is that the verb must have its subject and be marked [*subj*:+]. The rules for QUI and QUE could be merged assuming that *qui* and *que* are two forms of a same lexeme with an opposition of case (nominative for *qui* and oblique for *que*) (Kahane 2002). Moreover the two rules for QUE ($D_{que-relative}$ and $D_{que-complement}$ in Fig. 20) are almost identical: only the syntactic functions involved change. This corroborates

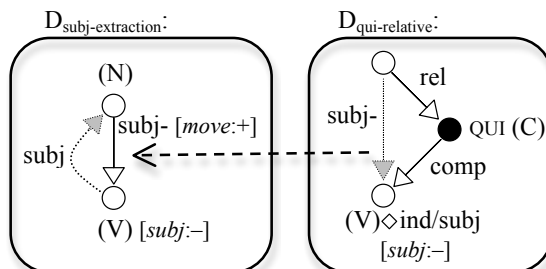Kayne's hypothesis that all these complementizers are one and the same.
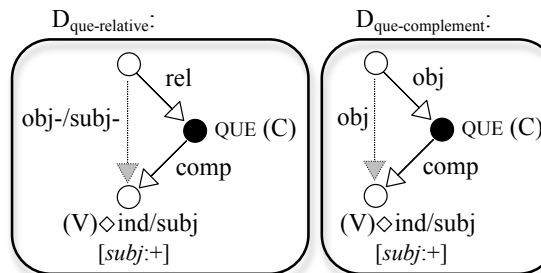

Figure 19. Rules for qui-relatives

Figure 20. Rules for que-clauses

6   Conclusion

We have presented a dependency grammar that constructs syntactic dependency trees like any other dependency grammar. But this grammar accomplishes more than others. First, it takes into account the syntax-semantics interface: the derivation structure can be interpreted as a graph of predicate-argument relations, that is, as the skeleton of a semantic representation (in the sense of Mel'čuk 1988, 2012). Second, this grammar accomodates the syntax-text interface, that is, linearization and morphology. Even though the linearization rules are not presented here (see Kahane & Lareau 2005 or Gerdes 2004 for linearization rules in PUG), it could be shown that the derived structure contains all the surface syntactic dependencies necessary to calculate the linear order (Gerdes & Kahane 2001).

In other respects, the grammar presented here is very modular. Grammatical constructions have their own independent rules, separated from the lexical rules, which describe only the base construction of lexemes. From this point a view, this grammar enters in the paradigm of construction grammars (CxGs) and can even be seen as a formalization of the Relational Grammar (Perlmutter 1980).

Our formalism makes a big use of the invisible polarity and the predicative adjunction. Formally, the derived structure is a graph with

145

a tree skeleton, which is the visible part of the derived structure (the objects polarized in black). The fact that we constantly maintain a tree skeleton during the derivation process is probably an important property from a computational point of view, but this has not been investigated further and the formalism has not been implemented.

Acknowledgments

References

Abney S. 1987. *The English Noun Phrase in its Sentential Aspect*. PhD Dissertation, MIT.

Candito M.-H., Kahane S. 1998. Can the derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory, *Proc. TAG+4*, Philadelphia, 21-24.

Candito M.-H. 1996. A principle-based hierarchical representation of LTAGs. *Proc. COLING*, Copenhagen.

Fillmore Ch. J. 1968. The Case for Case. In Bach and Harms (eds.): *Universals in Linguistic Theory*. Holt, Rinehart, and Winston, New York, 1-88.

Gerdes K. 2004. Tree Unification Grammar Problems and Proposals for Topology, TAG, and German, *Electronic Notes in Theoretical Computer Science*, 53, 108-136.

Gerdes K., Kahane S. 2001. Word Order in German: A Formal Dependency Grammar Using a Topological Hierarchy. *Proc. ACL*, Toulouse.

Hudson R. 2007. *Language Networks: The New Word Grammar*, Oxford University Press.

Joshi A. 1987. Introduction to Tree Adjoining Grammar. In Manaster Ramer (ed.), *The Mathematics of Language*, Benjamins, Amsterdam, 87-114.

Kahane S. 2001. A fully lexicalized grammar for French based on Meaning-Text theory. *Proc. CICLing*, Mexico, Springer Verlag, 18-31.

Kahane S. 2002. A propos de la position syntaxique des mots qu-. In P. Le Goffic (ed.), *Interrogation, indéfinition, subordination*, Verbum, 24(4), 399-435.

Kahane S. 2006. Polarized Unification Grammars, *Proc. Coling-ACL,* Sydney.

Kahane S. 2009. On the Status of Phrases in Head-driven Phrase Structure Grammar: Illustration by a Fully Lexical Treatment of Extraction, in A. Polguère & I. Mel'čuk (eds.), *Dependency in Linguistic Description*, Benjamins,111-150.

Kahane S., Mel'čuk I. 1999. La synthèse sémantique ou la correspondance entre graphes sémantiques et arbres syntaxiques – Le cas des phrases à extraction en français contemporain, *T.A.L.*, 40(2), 25-85.

Kahane S., Lareau F. 2005. Meaning-Text Unification Grammar: Modularity and polarization. *MTT 2005*, Moscow.

Kaplan R., Zaenen A. 1989. Long-distance dependencies, constituent structure, and functional uncertainty. In M. Baltin and A. Kroch (eds.), *Alternative Conceptions of Phrase Structure.* Chicago University Press, 17-42.

Kayne R. 1975. *French Syntax: The Transformational Cycle*, Cambridge : MIT Press.

Kroch A., Joshi A. 1986. Analysing Extraposition in a TAG. In G. Huck & A. Ojeda (eds.), *Discontinuous Constituents*, Syntax and Semantics, vol. 20, Academic Press, 107-149.

Lareau F. 2008. Vers une grammaire d'unification Sens-Texte du français : le temps verbal dans l'interface sémantique-syntaxe, Ph.D. thesis, University of Montreal/University Paris 7.

Mel'čuk I. 1988. *Dependency Syntax: Theory and Practice*, SUNY, Albany.

Mel'čuk I. 2012*, Semantics: From Meaning to Text*, Benjamins, Amsterdam.

Nasr A. 1995. A Formalism and a Parser for Lexicalised Dependency Grammars, *Proc. IWPT*, Prague, 186-195.

Perlmutter D. 1980. Relational grammar. In E. Moravcsik & J. Wirth (eds.), *Syntax and semantics: Current approaches to syntax* (Vol. 13), Academic Press, New York, 195-229.

Pollard Carl, Sag Ivan, 1994, *Head-Driven Phrase Structure Grammar*, CSLI, Stanford.

Rambow O., Vijay-Shanker K., Weir D. 1995. D-tree grammars. *Proceedings of ACL,* Cambridge.

Tesnière L. 1959. *Éléments de syntaxe structurale.* Klincksieck, Paris.

Rambow O., Joshi A. 1994. A formal look at dependency grammars and phrase-structure grammars, with special consideration of word-order phenomena. In L. Wanner (ed.), *Current Issue in Meaning-Text Theory*, Pinter,London.

Ross J. 1967. *Constraints on Variables in Syntax.* PhD Thesis, MIT; 1985. *Infinite syntax !*, Reidel, Dordrecht.

Shieber S., Schabes Y. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics,* 20(1), 91-124.

Vijay-Shanker K. 1987. *A Study of Tree Adjoining Grammars*. PhD thesis, University of Pennsylvania.