On the Semantics of Dependencies: Relative Clauses and Open Clausal Complements

Philippe de Groote

On the Semantics of Dependencies: Relative Clauses and Open Clausal Complements

Philippe de Groote(✉)

Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France
degroote@loria.fr

Abstract. In a previous work [4], we laid the foundations of a formal compositional semantic theory for dependency grammars. In this paper, we continue this line of research with the aim of showing that the basic principles we stated in [4] allow one to deal with more advanced syntactic phenomena. We consider two cases: the relative clauses (which depend on the acl:relcl dependency relation) and the open clausal complements (which depend on the xcomp dependency relation). This leads us to revise some of the solutions advocated in [4], while at the same time generalizing some of the principles on which our theory is based.

AQ1

1 Introduction

Dependency grammars, which stem from a long linguistic tradition [9,14], have seen a resurgence of interest in recent years, especially due to the Universal Dependency project [11]. Their principle is to make explicit the dependency relations existing between the words of a sentence, these relations being identified by the syntactic functions they represent.

Dependency parsing offers an interesting alternative to constituency parsing. In particular, parsing an agrammatical sentence does not result in a failure, but in a partial dependency structure that still contains some information. For this reason, dependency parsing is considered to be more robust than constituent parsing.

Formal compositional semantics, in the tradition of Montague [10], is based on a homomorphism between syntactic structure and semantic representation. As a result, it relies heavily on the notion of constituent, which does not appear explicitly in dependency structures. For this reason, providing a dependency grammar with a Montagovian semantic interpretation is not straightforward.

Nonetheless, in recent years, several ways of adapting Montague's semantics to the case of dependency grammars have been proposed in the literature. In a series of papers, Haug and his co-authors show how to assign a formal interpretation to a dependency structure using glue semantics [5–8]. Other authors, including ourself, rely on a compositionality principle close to Montague's by exploiting a functional representation of dependency structures [12,13]. This is the path we took in a previous paper [4], in which we established possible foundations for a formal semantic theory for dependency grammars. In the present

D. Bekki et al. (Eds.): LENLS 2023, LNCS 14569, pp. 1–16, 2024.
https://doi.org/10.1007/978-3-031-60878-0_14

paper, we continue this line of thought with the intention of showing that the approach we outlined in [4] allows for the treatment of more advanced linguistic phenomena. In particular, we study the cases of two dependency relations: acl:rel, which subordinates a relative clause to a noun, and xcomp, which subordinates an open clausal complement to a control verb. This study then leads us to revise some of the positions we took in [4].

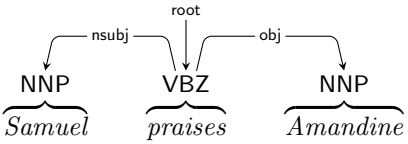
The rest of the paper is organized as follows. In the next section, we present the foundational principles underlying the semantic treatment of dependencies that we advocated in [4]. In Sect. 3, we address the problems associated with wh-extraction and the semantic interpretation of relative clauses. In Sect. 4, we discuss another linguistic construct, that of open clausal complements. These are clausal complements without their own subject, and whose semantic subject is typically provided by a complement (subject or object) of the control verb on which they depend. In Sect. 5, we discuss the solutions proposed in the previous two sections and explain why they are not entirely satisfactory. This leads us to revise our treatment of verbs in order to take into account certain elements of lexical semantics, in particular subcategorization. In Sect. 6, we bring together the various elements discussed in the previous sections and propose an integrated solution to the different problems we have addressed. We illustrate this solution with a small grammar that provides a unified treatment of the various linguistic phenomena we have considered. Finally, in Sect. 7, we conclude.

2 Foundational Principles of a Semantic Theory of Syntactic Dependencies

In [4], we laid the foundations for a possible semantic theory of syntactic dependencies. Let us synthesize and state the basic principles that outline this theory. To this end, consider the following simple sentence, together with its dependency struture:[1]

(1) *Samuel praises Amandine*

(2)



In order to take advantage of structure (2) with the goal of assigning a semantic representation to sentence (1), we need to address the issue of the syntax-semantic interface. When dealing with constituency grammars (typically, phrase

[1] The dependency structures occurring in this paper were obtained using the Stanford parser (online version: https://corenlp.run/). In these structures, labels such as NNP or VBZ are just part-of-speech tags. They should not be confused with the syntactic categories such as NP, VP, S, ... that we will use to type terms that encode dependency structures.

structure grammars or categorial grammars), a neat solution to the syntax-semantic interface problem is to represent syntactic structures as simply typed λ-terms [15]. In order to adapt this approach to the present case, we consider dependency relations as binary functions that take as arguments the governor and the governee of the relation. This allows structure (2) to be represented by the following term:

(3) $\mathsf{root}\,(\mathsf{nsubj}\ \textsc{samuel}\,(\mathsf{obj}\ \textsc{praises}\ \textsc{amandine}))$

Term (3) needs to be well-typed, and its typing derivation then corresponds to a dependency parsing of sentence (1):

$$\dfrac{\dfrac{\dfrac{\left.\dfrac{\mathsf{nsubj} : \textsc{np} \to \textsc{vp} \to \textsc{vp} \quad \textsc{samuel} : \textsc{np}}{\mathsf{nsubj}\ \textsc{samuel} : \textsc{vp} \to \textsc{vp}}\right\} \Pi_0 \quad \left.\dfrac{\mathsf{obj} : \textsc{vp} \to \textsc{np} \to \textsc{vp} \quad \textsc{praises} : \textsc{vp}}{\mathsf{obj}\ \textsc{praises} : \textsc{np} \to \textsc{vp}}\right\} \Pi_1}{\mathsf{nsubj}\ \textsc{samuel} : \textsc{vp}} \quad \dfrac{\vdots\ \Pi_1}{\mathsf{obj}\ \textsc{praises} : \textsc{np} \to \textsc{vp} \quad \textsc{amandine} : \textsc{np}}}{\mathsf{root} : \textsc{vp} \to \textsc{s}} \quad \dfrac{\vdots\ \Pi_0}{\mathsf{nsubj}\ \textsc{samuel} : \textsc{vp} \to \textsc{vp}} \quad \dfrac{\mathsf{obj}\ \textsc{praises} : \textsc{np} \to \textsc{vp} \quad \textsc{amandine} : \textsc{np}}{\mathsf{obj}\ \textsc{praises}\ \textsc{amandine} : \textsc{vp}}}{\mathsf{root}\,(\mathsf{nsubj}\ \textsc{samuel}\,(\mathsf{obj}\ \textsc{praises}\ \textsc{amandine})) : \textsc{s}}$$

The above derivation, however, is not the only possible parsing of sentence (1). Indeed, there exists another derivation:

$$\dfrac{\dfrac{\dfrac{\left.\dfrac{\mathsf{nsubj} : \textsc{np} \to \textsc{vp} \to \textsc{vp} \quad \textsc{samuel} : \textsc{np}}{\mathsf{nsubj}\ \textsc{samuel} : \textsc{vp} \to \textsc{vp}} \quad \textsc{praises} : \textsc{vp}\right\} \Pi}{\mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises} : \textsc{vp}}}{\mathsf{obj} : \textsc{vp} \to \textsc{np} \to \textsc{vp} \quad \mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises} : \textsc{vp}} \quad \dfrac{\vdots\ \Pi}{\mathsf{obj}\,(\mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises}) : \textsc{np} \to \textsc{vp}} \quad \textsc{amandine} : \textsc{np}}{\mathsf{root} : \textsc{vp} \to \textsc{s}} \quad \dfrac{\mathsf{obj}\,(\mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises})\ \textsc{amandine} : \textsc{vp}}{\mathsf{root}\,(\mathsf{obj}\,(\mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises})\ \textsc{amandine}) : \textsc{s}}$$

This second derivation yields another encoding of structure (2):

(4) $\mathsf{root}\,(\mathsf{obj}\,(\mathsf{nsubj}\ \textsc{samuel}\ \textsc{praises})\ \textsc{amandine})$

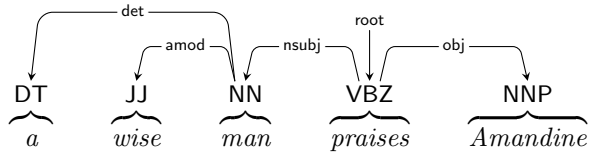
Nonetheless, sentence (1) is not semantically ambiguous. The existence of two different parsings therefore corresponds to a spurious ambiguity. In [4], in order to overcome this difficulty, we stated a coherence principle according to which the semantic interpretation of both terms (3) and (4) must yield the same result.

As a consequence of the coherence principle, every dependency relation must be assigned a type of the form $\alpha \to \beta \to \alpha$ (or $\beta \to \alpha \to \alpha$),[2] where α is the type of the governor and β the type of the governee.

[2] Here, we depart slightly from [4] by accepting both type schemes for dependency relations. This allows the order of the arguments of a dependency relation to reflect the word order, which makes terms like (3) and (4) more readable.

Now that we have a type scheme for assigning a syntactic category to a dependency relation, the next question we need to answer is: what should be the semantic interpretation of the syntactic categories? With this question in mind, consider sentence (5), here below, with its dependency structure:

(5) *a wise man praises Amandine*

(6)


In accordance with the typing discipline we have adopted, all the phrases *man*, *wise man*, *a wise man*—or more precisely, their encodings as λ-terms, i.e., MAN, amod WISE MAN, det A (amod WISE MAN)— must be assigned the same syntactic type, namely NP, and consequently the same semantic type. This is in conflict with the usual Montagovian approach, where *man* and *wise man* are assigned the semantic type $\mathbf{e} \to \mathbf{t}$, while *a wise man* is assigned the semantic type $(\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$. The way we propose to resolve this conflict is to give to a phrase its usual Montagovian interpretation, parameterized by the interpretations of all the potential governees of the head of that phrase. Let us illustrate this principle by considering the case of noun phrases. As mentioned above, the usual interpretation of the syntactic category of noun phrases is the type $(\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$. Now let us consider (as a simplifying hypothesis) that the only possible governees of the head of a noun phrase are determiners (with $(\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$ as semantic type) and adnominal modifiers (with $(\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}$ as semantic type). Accordingly, in our setting, the semantic type we assign to the syntactic category of noun phrases is:

(7) $\text{NP} := \mathbf{det} \to \mathbf{adnom} \to \mathbf{np}$

$$\text{where: } \begin{cases} \mathbf{det} = (\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t} \\ \mathbf{adnom} = (\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t} \\ \mathbf{np} = (\mathbf{e} \to \mathbf{t}) \to \mathbf{t} \end{cases}$$

That is, unfolding all the abbreviations:

(8) $\text{NP} := ((\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}) \to ((\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$

Using the above semantic interpretation of the category NP, we can now give a piece of grammar[3] that allows the noun phrase *a wise man* to be given a semantic interpretation.

Abstract Syntax

[3] We present it in the form of an Abstract Categorial Grammar [3].

ADJ, DET, NP : *type* ;
MAN : NP ;
WISE : ADJ ;
A : DET ;
amod : ADJ → NP → NP ;
det : DET → NP → NP

Object Language

$\mathbf{e}, \mathbf{t}$: *type* ;
$\mathbf{man}, \mathbf{wise} : \mathbf{e} \to \mathbf{t}$

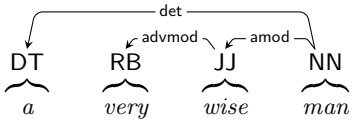
Semantic Interpretation

ADJ := $(\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}$;
DET := $(\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$;
NP := $((\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}) \to ((\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$;
MAN := $\lambda da.\, d\,(a\,\mathbf{man})$;
WISE := $\lambda nx.\, (n\,x) \wedge (\mathbf{wise}\,x)$;
A := $\lambda pq.\, \exists x.\, (p\,x) \wedge (q\,x)$;
amod := $\lambda andb.\, n\,d\,(\lambda x.\, b\,(a\,x))$;
det := $\lambda dne.\, n\,d$

A few comments are in order regarding the semantic interpretation above. For the adjective (WISE) and the determiner (A), the interpretations are as usual. This is because we consider that adjectives and determiners cannot be the governors of any dependency relation.[4] The interpretation of the noun (MAN) must be of type (7). Consequently, it takes two parameters, the λ-variables d and a. The purpose of the first parameter, d, is to be eventually instantiated by the semantic interpretation of the determiner of the noun phrase whose the noun *man* is the head. As for the second parameter, a, it plays the part of a possible adnominal modifier. Accordingly, the interpretation of the dependency relation det (respectively amod) allows the first argument (respectively the second argument) of the interpretation of a noun phrase to be given an actual value. For instance, interpreting the expression det A MAN yields the following λ-term:

(9) $\lambda eaq.\, \exists x.\, (a\,\mathbf{man}\,x) \wedge (q\,x)$

[4] This is clearly an over simplifying assumption. In particular, we do not take into account the possible adverbial modification of an adjective as in *a very wise man*:



.

Since det is of type $\textsc{det} \to \textsc{np} \to \textsc{np}$, the λ-term (9) must be of type (7). Thus, formally, the λ-variable e in (9) occupies the position of a determiner. But this determiner position, which is necessary for typing reasons, is in a sense fictive because the variable e does not occur in the body of (9). Similarly, the interpretation of a proper noun, which must also be of type (7), contains fictive parameters for the same typing reasons, resulting in vacuous lambda-abstractions. For instance, the interpretation of the proper noun *Amandine* is as follows:

(10) $\textsc{amandine} := \lambda dap.\, p\, \mathbf{amandine}$

where **amandine** is an object constant of type **e**.

Now, at some point in our semantic interpretation process, the noun phrases must be given their usual Montagovian interpretations. This can be done by providing the parameters of terms such as (9) or (10) with default values. To this end, we define the following saturating operator:

$$\mathrm{saturate}_{np} = \lambda n.\, n\, (\lambda pq.\, \exists x.\, (p\, x) \wedge (q\, x))\, (\lambda x.\, x)$$

Using this operator, we have that:

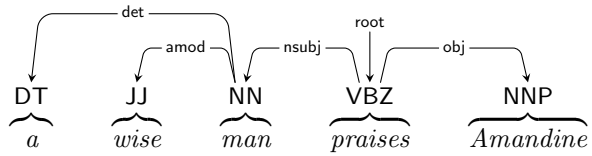
$$\mathrm{saturate}_{np}\, (\lambda eaq.\, \exists x.\, (a\, \mathbf{man}\, x) \wedge (q\, x)) \twoheadrightarrow_\beta \lambda q.\, \exists x.\, (\mathbf{man}\, x) \wedge (q\, x)$$
$$\mathrm{saturate}_{np}\, (\lambda dap.\, p\, \mathbf{amandine}) \twoheadrightarrow_\beta \lambda p.\, p\, \mathbf{amandine}$$

which corresponds to the usual Montagovian interpretation of the noun phrases *a man* and *Amandine*.

To finish this overview of the possible semantic theory whose first foundations we laid in [4], let us consider again sentence (5) and its dependency structure (6), which we repeat here as (11) and (12).

(11) *a wise man praises Amandine*

(12)



To complete the semantic treatment of (11), it remains to settle the question of verbs and the dependency relations they govern. At the abstract syntactic level, in line with our typing discipline, we need to extend our grammar as follows:

$\textsc{vp} : \mathit{type}$;
$\textsc{praises} : \textsc{vp}$;
$\mathsf{nsubj} : \textsc{np} \to \textsc{vp} \to \textsc{vp}$;
$\mathsf{obj} : \textsc{vp} \to \textsc{np} \to \textsc{vp}$

The semantic interpretation of the syntactic category $\textsc{vp}$ is then the next question to be dealt with. The answer we gave to this question in [4] is the one Champollion advocates in his treatment of event semantics [1]. It consists in

interpreting verb phrases as sets of sets of events. Following this line, we extend our grammar with the following declarations and interpretations:

$$\begin{array}{l}\mathbf{v} : \mathit{type}\,;\\ \mathbf{praise} : \mathbf{v} \to \mathbf{t}\,;\\ \mathbf{agent}, \mathbf{theme} : \mathbf{v} \to \mathbf{e} \to \mathbf{t}\end{array}$$

$$\begin{array}{l}\textsc{vp} := (\mathbf{v} \to \mathbf{t}) \to \mathbf{t}\,;\\ \textsc{praises} := \lambda p.\, \exists e.\, (\mathbf{praise}\, e) \wedge (p\, e)\,;\\ \mathsf{nsubj} := \lambda nvp.\, \mathrm{saturate}_{np}\, n\, (\lambda x.\, v\, (\lambda e.\, (\mathbf{agent}\, e\, x) \wedge (p\, e)))\,;\\ \mathsf{obj} := \lambda vnp.\, \mathrm{saturate}_{np}\, n\, (\lambda x.\, v\, (\lambda e.\, (\mathbf{theme}\, e\, x) \wedge (p\, e)))\end{array}$$

It is interesting to note that the relations that govern the noun phrases (i.e., nsubj and obj) operate their saturation. In a similar vein, we can define the root operator as a saturation operator for verb phrases.

$$\begin{array}{l}\textsc{s} : \mathit{type}\,;\\ \mathsf{root} : \textsc{vp} \to \textsc{s}\end{array}$$

$$\begin{array}{l}\textsc{s} := \mathbf{t}\,;\\ \mathsf{root} := \lambda v.\, \mathrm{saturate}_{vp}\, v,\ \text{where } \mathrm{saturate}_{vp} = \lambda v.\, v\, (\lambda x.\, \mathbf{true})\end{array}$$

Using the apparatus we have developed so far, we can give a semantics to sentence (11) by interpreting the following term:

(13) $\mathsf{root}\,(\mathsf{nsubj}\,(\mathsf{det}\,\textsc{a}\,(\mathsf{amod}\,\textsc{wise man}))\,(\mathsf{obj}\,\textsc{praises amandine}))$

which results in the following formula:

$$\exists x.\, (\mathbf{man}\, x) \wedge (\mathbf{wise}\, x) \wedge (\exists e.\, (\mathbf{praise}\, e) \wedge (\mathbf{agent}\, e\, x) \wedge (\mathbf{theme}\, e\, \mathbf{amandine}))$$

Let us conclude this section dedicated to the principles underlying our semantic theory by summarizing them.

1. Dependency relations are represented as binary functions of type $\alpha \to \beta \to \alpha$ (or $\beta \to \alpha \to \alpha$), where α is the syntactic category of the governor and β is the syntactic category of the governee.
2. By virtue of a coherence principle, the different ways of encoding a dependency structure by means of a λ-term should all give rise to the same semantic interpretation.
3. The semantic interpretation of a syntactic category CAT is a type of the form $\beta_1 \to \cdots \beta_n \to \alpha$, where α is the Montagovian interpretation of CAT and $\beta_1, \ldots \beta_n$ are the Montagovian interpretations of the syntactic categories of the phrases whose heads can potentially be governed by the head of a phrase of category CAT.
4. Saturating operators allow phrases to recover their usual Montagovian interpretations.
5. Verbs and verbal phrases are semantically interpreted as sets of sets of events.

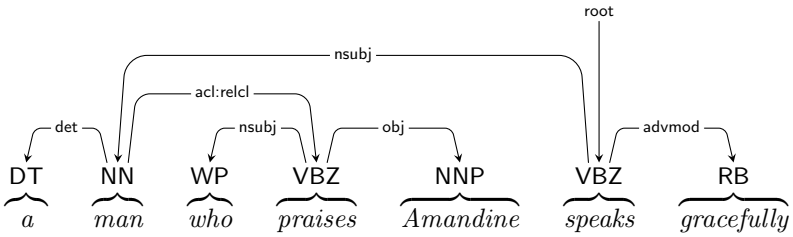
3 Relative Clauses and Wh-Extraction

In this section, we tackle the problems associated with the semantic treatment of wh-extraction, a phenomenon we did not address in [4]. As a paradigmatic example, we deal with the case of relative clauses.

Consider the following sentence together with its dependency structure and its abstract syntax.

(14) *a man who praises Amandine speaks gracefully*

(15)



(16) root (nsubj(det A (acl:relcl MAN (nsubj WHO (obj PRAISES AMANDINE)))) (advmod SPEAKS GRACEFULLY))

The new ingredients here are the relative pronoun WHO and the dependency relation acl:relcl.[5] According to our typing discipline, acl:relcl must be typed as follows:

$$\mathsf{acl{:}relcl} : \text{NP} \to \text{WVP} \to \text{NP}$$

where WVP is the type of the relative clauses. This being established, let us take a look at what might be an appropriate semantic interpretation of the type WVP. Compared to a simple clause, a relative clause has the specificity of having undergone a wh-movement. From a semantic point of view, such a movement can be modeled as a λ-abstraction. Typically, if p is a phrase of category CAT that has undergone a wh-extraction, and if **cat** is the semantic type associated with CAT, then the semantic interpretation of p will be of type $\mathbf{e} \to \mathbf{cat}$. Applying this to our setting, we have that

$$\text{VP} := (\mathbf{v} \to \mathbf{t}) \to \mathbf{t}$$

and therefore that

$$\text{WVP} := \mathbf{e} \to (\mathbf{v} \to \mathbf{t}) \to \mathbf{t}$$

Let us now turn to the problem of interpreting acl:relcl. We expect the saturated interpretation of the relative clause *who praises Amandine* to be as follows:

(17) $\lambda x.\, \exists e.\, (\mathbf{praise}\, e) \wedge (\mathbf{agent}\, e\, x) \wedge (\mathbf{theme}\, e\, \mathbf{amandine})$

[5] Neither did we consider the adverbial modification of a verb phrase in the previous section. But this is not a problem, for it can be easily accommodated with the following declarations and interpretations: ADV : *type* ; GRACEFULLY : ADV ; advmod : VP → ADV → VP ; **graceful** : $\mathbf{v} \to \mathbf{t}$; ADV := $\mathbf{v} \to \mathbf{t}$; GRACEFULLY := $\lambda e.$ **graceful** e ; advmod := $\lambda v a f.\, v\, (\lambda e.\, (a\, e) \wedge (f\, e))$.

On the other hand, we have that the interpretation of the noun phrase *man* is:

(18) $\lambda da.\, d\,(\lambda x.\, a\,\mathbf{man}\,x)$

and we expect the interpretation of the noun phrase *man who praises Amandine* to be:

(19) $\lambda da.\, d\,(\lambda x.\, (a\,\mathbf{man}\,x) \wedge (\exists e.\, (\mathbf{praise}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (\mathbf{theme}\,e\,\mathbf{amandine})))$

So the question is: how can we combine (17) and (18) to obtain (19)? By working it out, we get the following solution:

$$\mathsf{acl{:}relcl} := \lambda nrda.\, n\,d\,(\lambda nx.\, (a\,n\,x) \wedge (\mathrm{saturate}_{vp}\,(r\,x)))$$

In a related line of thought, we get the following interpretation of the relative pronoun:

$$\begin{array}{l}\textsc{wnp} : \mathit{type}\,;\\ \textsc{wnp} := \mathbf{e} \to ((\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}) \to ((\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}\,;\\ \textsc{who} : \textsc{wnp}\,;\\ \textsc{who} := \lambda xdap.\, p\,x\end{array}$$

But then, the expression below gives rise to a type mismatch:

$$\mathsf{nsubj}\ \textsc{who}\ \textsc{praises}$$

which implies that term (16) is not well typed. Indeed, nsubj is of type $\textsc{np} \to \textsc{vp} \to \textsc{vp}$ while who is of type wnp. A possible way out of this problem would be to give nsubj a second interpretation that would be of the appropriate type:

$$\begin{array}{l}\textsc{nsubj} : \textsc{wnp} \to \textsc{vp} \to \textsc{wvp}\,;\\ \textsc{nsubj} := \lambda nvpx.\, \mathrm{saturate}_{np}\,(n\,x)\,(\lambda x.\, v\,(\lambda e.\, (\mathbf{agent}\,e\,x) \wedge (p\,e)))\end{array}$$

This solution, however, is not entirely satisfactory. On the one hand, it overloads the relation nsubj, for which we would prefer to keep a unique interpretation. On the other hand, it does not obey the type scheme we have adopted for the dependency relations, which results in a violation of our coherence principle.

Another workaround is to use operators to coerce the types of the dependency relations, which is similar to the way combinators are used in combinatorial logic to simulate λ-abstraction [2]:

$$\begin{array}{l}\mathsf{carg}_1 : (\textsc{np} \to \textsc{vp} \to \textsc{vp}) \to \textsc{wnp} \to \textsc{vp} \to \textsc{wvp}\,;\\ \mathsf{carg}_1 := \lambda rnvx.\, r\,(n\,x)\,v\end{array}$$

Then, the semantic interpretation of the noun phrase, *man who praises amandine*, may be computed by interpreting the following expression:

$$\mathsf{acl{:}relcl}\ \textsc{man}\ (\mathsf{carg}_1\ \mathsf{nsubj}\ \textsc{who}\ (\mathsf{obj}\ \textsc{praises}\ \textsc{amandine}))$$

which results in the following λ-term:[6]

$$\lambda da.\, d\,(\lambda x.\, (a\,\mathbf{man}\,x) \wedge (\exists e.\, (\mathbf{praise}\,e) \wedge (\mathbf{theme}\,e\,\mathbf{amandine}) \wedge (\mathbf{agent}\,e\,x)))$$

To make this solution compatible with the coherence principle, we can now define another coercion operator:

$$\begin{aligned}&\mathsf{carg}_2 : (\textsc{vp} \to \textsc{np} \to \textsc{vp}) \to \textsc{wvp} \to \textsc{np} \to \textsc{wvp}\,;\\ &\mathsf{carg}_2 := \lambda rvnx.\, r\,(v\,x)\,n\end{aligned}$$

It allows the dependency structure of sentence (14) to be encoded by another term:

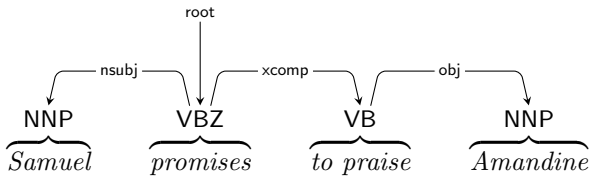
$$\mathsf{acl{:}relcl}\ \textsc{man}\ (\mathsf{carg}_2\ \mathsf{obj}\ (\mathsf{carg}_1\ \mathsf{nsubj}\ \textsc{who praises})\ \textsc{amandine})$$

whose interpretation yields the following λ-term:

$$\lambda da.\, d\,(\lambda x.\, (a\,\mathbf{man}\,x) \wedge (\exists e.\, (\mathbf{praise}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (\mathbf{theme}\,e\,\mathbf{amandine})))$$

4 Control Verbs and Open Clausal Complements

In this section, we turn to another linguistic phenomenon, that of open clausal complements. Such complements are clauses that are missing their own subject and that are typically governed by a control verb or a raising verb.[7] The dependency relation between a control verb and its open clausal complement is marked by the xcomp relation, like in the following example:



whose expected semantic interpretation is as follows:

$$\begin{aligned}&\exists e.\, (\mathbf{promise}\,e) \wedge\\ &\quad (\mathbf{agent}\,e\,\mathbf{samuel}) \wedge\\ &\quad (\mathbf{topic}\,e\,(\exists e.\, (\mathbf{praise}\,e) \wedge (\mathbf{agent}\,e\,\mathbf{samuel}) \wedge (\mathbf{theme}\,e\,\mathbf{amandine})))\end{aligned}$$

The puzzle here is that the nsubj dependency relation must provide a subject not only for *promises* but also for *praise*. This means that the interpretation of the verb phrase *to praise Amandine* must be parameterized by an entity variable that will be intantiated by a value provided by the interpretation of the verb phrase *Samuel promises*. A technical way of achieving this is based on a new interpretation of verb phrases that is obtained by type-raising the type of events not on $\mathbf{t}$ but on $\mathbf{e} \to \mathbf{t}$:

[6] Modulo the following elmentary logical equivalence: $\alpha \wedge \mathbf{true} \equiv \alpha$.

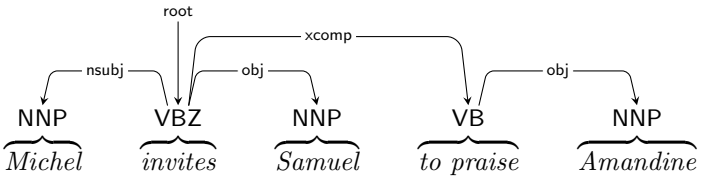
[7] An open clausal complement may also be the governee of an adjective.

(20) $\text{VP} = (\mathbf{v} \to \mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}$

This idea, leads to the following interpretation of xcomp together with a revisited interpretation of nsubj to be used for control verbs:

$$\begin{aligned}\mathsf{xcomp} &:= \lambda vcpx.\, v\,(\lambda ey.\, (\mathbf{topic}\, e\, (c\, (\lambda fz.\, \mathbf{agent}\, f\, z)\, y)) \wedge (p\, e\, y))\, x \\ \mathsf{nsubj} &:= \lambda vnpx.\, \text{saturate}_{np}\, n\, (\lambda y.\, v\, (\lambda ez.\, (\mathbf{agent}\, e\, y) \wedge (p\, e\, y))\, y)\end{aligned}$$

But this is not the end of the story, because it is not always the subject of the control verb that provides the open clausal complement with a semantic subject. It can also be the object, as the following sentence illustrates:



Consequently, we also need a revisited interpretation of obj to be used for control verbs akin to *invites*:

$$\mathsf{obj} := \lambda vnpx.\, \text{saturate}_{np}\, n\, (\lambda y.\, v\, (\lambda ez.\, (\mathbf{theme}\, e\, y) \wedge (p\, e\, y))\, y)$$

But the situation is even more intricate, as it is the case that the same control verb admits both constructions:

Michel wants to praise Amandine
Michel wants Samuel to praise Amandine

A solution based on (20) that can account for these phenomena is possible. However, it would be rather complicated, a point we discuss in the next section.

5 Taking Stock: The Question of Lexical Semantics

In Sect. 4, we said that we prefer a dependency relation to have a unique interpretation. With the solution we discussed in the previous section, this is no longer the case. Indeed, if we adopt the solution advocated in the previous section, we would need two interpretations of the relation nsubj.

In fact the situation is even worse. In what we have developed so far, it is the dependency relations that introduce the thematic roles. However, it is not the case that there is a one-to-one correspondence between syntactic functions and semantic roles. A subject, for example, can correspond to an **agent** or an **experiencer**. Similarly, an object can correspond to a **patient**, a **theme**, a **recipient**, or even a **stimulus**. As a result, if we stick to the approach we have

taken so far, we would face a combinatorial explosion of the possible interpretations of a same dependency relation. What it seems is that we have made the mistake of encoding information that is part of lexical semantics within the dependency relations.

It would be more appropriate to have for the subcategorization of a verb to appear in its lexical entry, as in the following examples:

$$\begin{aligned}
&\textsc{speak} := \lambda xp.\, \exists e.\, (\mathbf{speak}\, e) \wedge (\mathbf{agent}\, e\, x) \wedge (p\, e)\\
&\textsc{praise} := \lambda xyp.\, \exists e.\, (\mathbf{praise}\, e) \wedge (\mathbf{agent}\, e\, x) \wedge (\mathbf{theme}\, e\, y) \wedge (p\, e)\\
&\textsc{like} := \lambda xyp.\, \exists e.\, (\mathbf{like}\, e) \wedge (\mathbf{experiencer}\, e\, x) \wedge (\mathbf{stimulus}\, e\, y) \wedge (p\, e)
\end{aligned}$$

But then, the possible problem is that we no longer have a unique semantic type for the category of verb phrases. The cure for this possible defect is to base the interpretation of the verb phrases on the same principle on which the interpretation of the noun phrases is based, that is to parameterize the semantic type of the verb phrases with the types of all the possible governees of a verb. In the fragment we are dealing with the verbs have four kinds of governees: subjects ($\mathbf{e}$), objects ($\mathbf{e}$), open clausal complements ($\mathbf{e} \to \mathbf{t}$), and adverbial modifiers ($\mathbf{v} \to \mathbf{t}$). Consequently, we obtain the following interpretation:

$$\textsc{vp} := \mathbf{e} \to \mathbf{e} \to (\mathbf{e} \to \mathbf{t}) \to (\mathbf{v} \to \mathbf{t}) \to \mathbf{t} \tag{21}$$

With this interpretation, the lexical entry of the verb *praise* becomes the following one:

$$\textsc{praise} := \lambda xycp.\, \exists e.\, (\mathbf{praise}\, e) \wedge (\mathbf{agent}\, e\, x) \wedge (\mathbf{theme}\, e\, y) \wedge (p\, e)$$

where the λ-variable c is a dummy parameter that does not occur in the interpretation of the verb and is present only for typing reasons.

Because of the presence of dummy parameters, we need to define an appropriate saturating operator for the verb phrases. We do this by using existential closures:

$$\mathrm{saturate}_{vp} = \lambda v.\, \exists xy.\, v\, x\, y\, (\lambda x.\, \mathbf{true})\, (\lambda e.\, \mathbf{true})$$

Using interpretation (21) greatly simplifies the interpretations of the dependencies. For example, the interpretations of nsubj and xcomp, which are now unique, are as follows:

$$\begin{aligned}
&\mathsf{nsubj} := \lambda nvxycf.\, \mathrm{saturate}_{np}\, n\, (\lambda z.\, v\, z\, y\, c\, f)\\
&\mathsf{xcomp} := \lambda vcxydf.\, v\, x\, y\, (\lambda x.\, \mathrm{saturate}_{vp}\, (\lambda z.\, cx))\, f
\end{aligned}$$

It is worth noting that these new interpretations are pure combinators that do not introduce any lexical information.

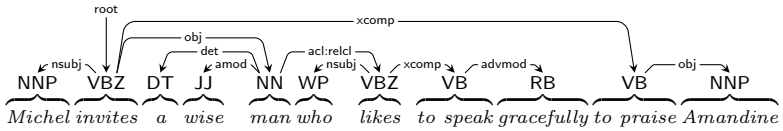
In the next section, we present a grammar based on interpretation (21).

6 Bringing Everything Together

The new interpretation of the verb phrases advocated in the previous section requires a revision of the solutions we discussed in Sects. 2 and 3. We do this in

this section by developing a toy grammar that integrates the various elements we have discussed. This grammar allows for the semantic treatment of a sentence like the following:

(22) *Michel invites a wise man who likes to speak gracefully to praise Amandine*



We present our integrated semantic grammar in the form of an Abstract Categorial Grammar.[8]

Abstract Syntax

ADJ, ADV, DET, NP, VP, WNP, WVP, S : *type* ;
A : DET ;
WISE : ADJ ;
MAN : NP ;
PRAISE, PROMISE, SPEAK, LIKE, INVITE : VP ;
AMANDINE, MICHEL, SAMUEL : NP ;
GRACEFULLY : ADV ;
WHO, WHOM : WNP ;
amod : ADJ ⟶ NP ⟶ NP ;
advmod : VP ⟶ ADV ⟶ VP ;
det : DET ⟶ NP ⟶ NP ;
nsubj : NP ⟶ VP ⟶ VP ;
obj : VP ⟶ NP ⟶ VP ;
xcomp : VP ⟶ VP ⟶ VP ;
acl:relcl : NP ⟶ WVP ⟶ NP ;
root : VP ⟶ S ;
wnparg1 : (NP ⟶ VP ⟶ VP) ⟶ WNP ⟶ VP ⟶ WVP ;
wnparg2 : (VP ⟶ NP ⟶ VP) ⟶ VP ⟶ WNP ⟶ WVP ;
wvparg1 : (VP ⟶ NP ⟶ VP) ⟶ WVP ⟶ NP ⟶ WVP ;
wvparg2 : (NP ⟶ VP ⟶ VP) ⟶ NP ⟶ WVP ⟶ WVP

Object Language

e, **t**, **v** : *type* ;
amandine, **michel**, **samuel** : **e** ;
man, **wise** : **e** ⟶ **t** ;
praise, **promise**, **speak**, **like**, **invite** : **v** ⟶ **t** ;

[8] In fact, we implemented it using the ACG toolkit: https://gitlab.inria.fr/ACG/dev/ACGtk..

$\mathbf{agent}, \mathbf{theme}, \mathbf{experiencer}, \mathbf{recipient} : \mathbf{v} \to \mathbf{e} \to \mathbf{t}$;
$\mathbf{topic}, \mathbf{stimulus} : \mathbf{v} \to \mathbf{t} \to \mathbf{t}$;
$\mathbf{graceful} : \mathbf{v} \to \mathbf{t}$;
$\mathrm{saturate}_{np} = \lambda n.\, n\,(\lambda pq.\, \exists x.\, (p\,x) \wedge (q\,x))\,(\lambda x.\, x)$;
$\mathrm{saturate}_{vp} = \lambda v.\, \exists xy.\, v\,x\,y\,(\lambda x.\, \mathbf{true})\,(\lambda e.\, \mathbf{true})$

Semantic Interpretation

ADJ $:= (\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}$;
ADV $:= \mathbf{v} \to \mathbf{t}$;
DET $:= (\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$;
NP $:= ((\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}) \to ((\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$;
VP $:= \mathbf{e} \to \mathbf{e} \to (\mathbf{e} \to \mathbf{t}) \to (\mathbf{v} \to \mathbf{t}) \to \mathbf{t}$;
WNP $:= \mathbf{e} \to ((\mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}) \to ((\mathbf{e} \to \mathbf{t}) \to \mathbf{e} \to \mathbf{t}) \to (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}$;
WVP $:= \mathbf{e} \to \mathbf{e} \to \mathbf{e} \to (\mathbf{e} \to \mathbf{t}) \to (\mathbf{v} \to \mathbf{t}) \to \mathbf{t}$;
S $:= \mathbf{t}$;
A $:= \lambda pq.\, \exists x.\, (p\,x) \wedge (q\,x)$;
WISE $:= \lambda nx.\, (n\,x) \wedge (\mathbf{wise}\,x)$;
MAN $:= \lambda da.\, d\,(a\,\mathbf{man})$;
SPEAK $:= \lambda xycp.\, \exists e.\, (\mathbf{speak}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (p\,e)$;
PRAISE $:= \lambda xycp.\, \exists e.\, (\mathbf{praise}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (\mathbf{theme}\,e\,y) \wedge (p\,e)$;
PROMISE $:= \lambda xycp.\, \exists e.\, (\mathbf{promise}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (\mathbf{topic}\,e\,(c\,x)) \wedge (p\,e)$;
LIKE $:= \lambda xycp.\, \exists e.\, (\mathbf{like}\,e) \wedge (\mathbf{experiencer}\,e\,x) \wedge (\mathbf{stimulus}\,e\,(c\,x)) \wedge (p\,e)$;
INVITE $:= \lambda xycp.\, \exists e.\, (\mathbf{invite}\,e) \wedge (\mathbf{agent}\,e\,x) \wedge (\mathbf{recipient}\,e\,y) \wedge (\mathbf{topic}\,e\,(c\,y)) \wedge (p\,e)$;
AMANDINE $:= \lambda dap.\, p\ \mathbf{amandine}$;
MICHEL $:= \lambda dap.\, p\ \mathbf{michel}$;
SAMUEL $:= \lambda dap.\, p\ \mathbf{samuel}$;
GRACEFULLY $:= \lambda e.\, \mathbf{graceful}\,e$;
WHO, WHOM $:= \lambda xdap.\, p\,x$;
amod $:= \lambda andb.\, n\,d\,(\lambda x.\, b\,(a\,x))$;
advmod $:= \lambda vaxycf.\, v\,x\,y\,c\,(\lambda e.\, (a\,e) \wedge (f\,e))$;
det $:= \lambda dne.\, n\,d$;
nsubj $:= \lambda nvxycf.\, \mathrm{saturate}_{np}\,n\,(\lambda z.\, v\,z\,y\,c\,f)$;
obj $:= \lambda vnxycf.\, \mathrm{saturate}_{np}\,n\,(\lambda z.\, v\,x\,z\,c\,f)$;
xcomp $:= \lambda vcxydf.\, v\,x\,y\,(\lambda x.\, \mathrm{saturate}_{vp}\,(\lambda z.\, c\,x))\,f$;
acl:relcl $:= \lambda nrda.\, n\,d\,(\lambda nx.\, (a\,n\,x) \wedge (\mathrm{saturate}_{vp}\,(r\,x)))$;
root $:= \lambda v.\, \mathrm{saturate}_{vp}\,v$;
wnparg1 $:= \lambda rnvx.\, r\,(n\,x)\,v$;
wnparg2 $:= \lambda rvnx.\, r\,v\,(n\,x)$;
wvparg1 $:= \lambda rvnx.\, r\,(v\,x)\,n$;
wvparg2 $:= \lambda rnvx.\, r\,n\,(v\,x)$

7 Conclusions

In this paper we have made explicit the principles underlying the semantic theory we began to develop in [4]. We have then shown how dependency relations involving implicit semantic arguments can be treated along these lines. This treatment, however, has necessitated a revision of the interpretation of the verb phrases. Quite satisfactorily, the new interpretation we have adopted generalizes the previous one, and is based on the principle we advocated for the interpretation of noun phrases.

It results in a semantic grammar in which the lexical semantics is encoded, as it should be, within the lexical entries. As for the dependency relations act, they act as pure combinators that assemble the meanings of the words that form a sentence. We believe that this reflects the spirit of dependency grammars and demonstrates the feasibility and adequacy of the formal theory of dependency semantics, we started to develop [4].

For future work, we intend to extend the coverage of our grammar by considering more linguistic constructs, and to conduct a real-size experiment. To this end, we will take advantage of the existence of numerous corpora annotated with dependencies, and attempt to built a semantic grammar semi-automatically from such a corpus.

References

1. Champollion, L.: The interaction of compositional semantics and event semantics. Linguist. Philos. **38**(1), 31–66 (2015)
2. Curry, H.B., Feys, R.: Combinatory Logic, vol. I, North-Holland (1958)
3. de Groote, P.: Towards abstract categorial grammars. In: Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference, pp. 148–155 (2001)
4. de Groote, P.: Deriving formal semantic representations from dependency structures. In: Bekki, D., Mineshima, K., McCready, E. (eds.) Logic and Engineering of Natural Language Semantics, LENLS 2022, LNCS, vol. 14213. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43977-3_10
5. Findlay, J.Y., Haug, D.T.T.: How useful are enhanced Universal Dependencies for semantic interpretation? In: Proceedings of the Sixth International Conference on Dependency Linguistics (Depling, SyntaxFest 2021), pp. 22–34. Association for Computational Linguistics (2021)
6. Findlay, J.Y., Salimifar, S., Yıldırım, A., Haug, D.T.T.: Rule-based semantic interpretation for Universal Dependencies. In: Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023), pp. 47–57. Association for Computational Linguistics (2023)
7. Haug, D.T.T., Findlay, J.Y.: Formal semantics for dependency grammar. In: Proceedings of the Seventh International Conference on Dependency Linguistics (Depling, GURT/SyntaxFest 2023), pp. 22–31. Association for Computational Linguistics (2023)
8. Haug, D.T.T., Gotham, M.: Glue semantics for universal dependencies. In: Proceedings of the LFG-conference, pp. 208–226. CSLI Publications (2021)

9. Mel'čuk, I.: Dependency Syntax: Theory and Practice. State University of New York Press, New York (1988)
10. Montague, R.: Formal Philosophy: selected papers of Richard Montague, edited and with an introduction by Richmond Thomason. Yale University Press, New Haven (1974)
11. Nivre, J., et al.: Universal Dependencies v1: a multilingual treebank collection. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), pp. 1659–1666, Portorož, Slovenia, European Language Resources Association (ELRA) May 2016
12. Reddy, S.: Transforming dependency structures to logical forms for semantic parsing. Trans. Assoc. Comput. Linguist. **4**, 127–140 (2016)
13. Reddy, S., Täckström, O., Petrov, S., Steedman, M., Lapata, M.: Universal semantic parsing. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 89–101. Association for Computational Linguistics (2017)
14. Tesnière, L.: Eléments de syntaxe structurale. Klincksieck, Paris (1959)
15. van Benthem, J.: Essays in Logical Semantics. Reidel, Dordrecht (1986)