Deriving formal semantic representations from dependency structures - Extended Abstract

Philippe de Groote

Deriving formal semantic representations from dependency structures
— extended abstract —

Philippe de Groote

LORIA, UMR 7503, Université de Lorraine, CNRS, Inria, 54000 Nancy, France

1 introduction

Dependency grammars provide an interesting alternative to phrase structure grammars. They derive from a long linguistic tradition [3], and are gaining more and more interest in the computational linguistic international community [6]. One of their advantages is that dependency parsing appears to be more robust than constituency parsing. Indeed, while parsing an agrammatical sentence with a phrase structure grammar usually leads to a failure, parsing it with a dependency grammar can result in an incomplete dependency structure that nevertheless carries some semantic information.

Dependency grammars, however, do not seem suitable for a formal semantic treatment, in the tradition of Montague [5]. Formal semantics [2], being compositional, relies heavily on the notion of constituent, a notion that does not appear explicitly within dependency structures.

A possible remedy to this this situation is to normalize the dependency structures in order to recover an implicite notion of constituent (see [7], for instance). This approach, however, is not robust in the sense that it does not allow for the interpretation of partial dependency structures. The goal of this paper is to remedy this problem by laying the grounds for a new formal theory of dependency semantics, in the spirit of Montague.

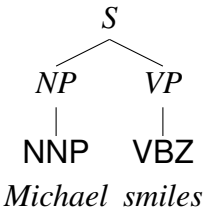
2 The basic concept

Consider the following simple sentence:

(1) *Michael smiles*

Parsing it with a phrase structure grammar would yield a constituency parse tree akin to the following one[1]:

[1] All the parse trees and the constituency structures occurring in this abstract have been obtained using the Stanford parser



Following Montague [4], one may take advantage of the above parse tree in order to derive the truth-conditional meaning of sentence (1). To this end, let us write S, NP, and VP for the semantic interpretations of the syntactic categories *S*, *NP*, and *VP*. As usual, one posits:

$$\begin{aligned} \text{S} &= \mathbf{t} \\ \text{VP} &= \mathbf{e} \to \mathbf{t} \\ \text{NP} = \text{VP} \to \text{S} &= (\mathbf{e} \to \mathbf{t}) \to \mathbf{t} \end{aligned}$$

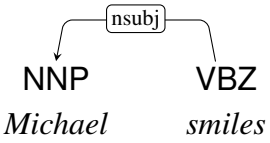
Then, one assigns the following lexical semantic recipes to *Michael* and *smiles*:

$$\begin{aligned} \text{MICHAEL} &= \lambda p.\, p\, \mathbf{m} : \text{VP} \to \text{S} \\ \text{SMILE} &= \lambda x.\, \mathbf{smile}\, x : \text{VP} \end{aligned}$$

This allows one to compute the the truth-conditional semantics of sentence (1) by reducing the corresponding λ-term:

(2) $\quad \text{MICHAEL SMILE} \twoheadrightarrow_\beta \mathbf{smile}\, \mathbf{m}$

Now, we want to apply the same kind of technique to dependency grammars, i.e, to derive the truth-conditional semantics of sentence (1) form the following dependency structure:

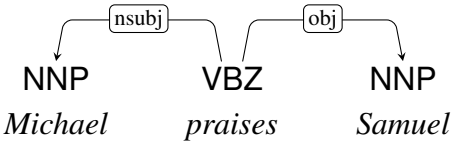


The solution we develop in this paper is based on a simple idea, which consists in assigning a semantic role to the dependency relations, and computing the desired semantics from a term akin to $\mathsf{nsubj}\,(\text{SMILE}, \text{MICHAEL})$ (or simply, nsubj SMILE MICHAEL, using the λ-calculus notation). For our current example, it suffices to let $\mathsf{nsubj} = \lambda vn.\, n\, v$. However, as we will see, this interpretation is too simple.

3 The coherence requirement

In pursuing the basic idea we sketched in the previous section, we soon run into an obstacle. There is indeed no canonical way of representing a dependency structure as a term. Consider, for instance, the following sentence together with its dependency structure:

(3) *Michael praises Samuel*



There is in fact two ways of encoding the above dependency structure as a term:

(4) a. $\mathsf{nsubj}\,(\mathsf{obj}\ \textsc{praise}\ \textsc{samuel})\ \textsc{michael}$
b. $\mathsf{obj}\,(\mathsf{nsubj}\ \textsc{praise}\ \textsc{michael})\ \textsc{samuel}$

One could try to circumvent this difficulty by preferring one of these representations to the other, but such a choice would be arbitrary. Moreover, the resulting solution would not be robust in the sense that it would not allow for the interpretation of partial dependency structures. Consequently, we require the coherence condition that both terms (4-a) and (4-b) must yield the same semantic interpretation.

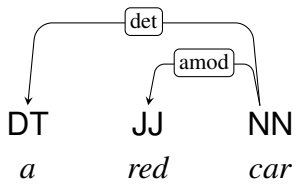
In order to satisfy this coherence condition, we adopt a Neo-Davisdsonian event semantics, and interpet a sentence as a set of sets of events, as suggested by Champolion [1]. This gives rise to the following semantic interpretation:

$$
\begin{aligned}
\textsc{gs} &= (\mathbf{v} \to \mathbf{t}) \to \mathbf{t} \\
\textsc{michael} &= \lambda p.\, p\,\mathbf{m} : \textsc{np} \\
\textsc{samuel} &= \lambda p.\, p\,\mathbf{s} : \textsc{np} \\
\textsc{praises} &= \lambda p.\, \exists e.\, (\mathbf{praise}\, e) \wedge (p\, e) : \textsc{gs} \\
\mathsf{nsubj} &= \lambda v n.\, \lambda p.\, n\,(\lambda x.\, v\,(\lambda e.\, (\mathbf{agent}\, e\, x) \wedge (p\, e))) : \textsc{gs} \to \textsc{np} \to \textsc{gs} \\
\mathsf{obj} &= \lambda v n.\, \lambda p.\, n\,(\lambda x.\, v\,(\lambda e.\, (\mathbf{theme}\, e\, x) \wedge (p\, e))) : \textsc{gs} \to \textsc{np} \to \textsc{gs}
\end{aligned}
$$

4 Interpreting the noun phrases

Montagovian semantics assigns to the (common) nouns the semantic category $\textsc{n} = \mathbf{e} \to \mathbf{t}$. It assigns to the adnominal modifiers, such as the adjectives, the category $\textsc{adj} = \textsc{n} \to \textsc{n}$, and to the determiners, the category $\textsc{det} = \textsc{n} \to \textsc{np}$. This approach is not directly transferable to the case of dependency structures. Consider indeed the folowing noun phrase and its associated dependency structure:

(5) *a red car*



As a consequence of the coherence condition, the following expressions must be assigned the same semantic type:

(6) a. CAR
b. amod CAR RED
c. det CAR A
d. det (amod CAR RED) A
e. amod (det CAR A) RED

More generally, if dep is the semantic recipe associated to a dependency edge (dep), we must have:

$$\mathsf{dep} : \alpha \to \beta \to \alpha$$

where α is the semantic type assigned to the source of the edge, and β, the semantic type assigned to its target.

A way of satisfying the above requirement is to parametrize the type assigned to the head of a dependency relation with the types assigned to all its possible dependents. In the case of the expressions listed in (6), this type is then the following one:

$$\text{GNP} = \text{DET} \to \text{ADJ} \to \text{NP}$$

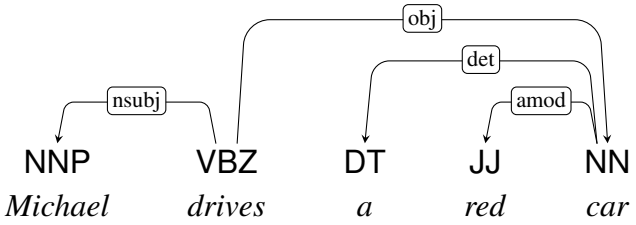
Accordingly, the semantic recipes associated to the lexical items and dependency relations are as follows:

$$\begin{aligned}
\text{A} &= \lambda pq.\, \exists x.\, (p\, x) \wedge (q\, x) : \text{DET} \\
\text{RED} &= \lambda nx.\, (n\, x) \wedge (\mathbf{red}\, x) : \text{ADJ} \\
\text{CAR} &= \lambda da.\, d\, (a\, \mathbf{car}) : \text{GNP} \\
\mathsf{amod} &= \lambda na.\, \lambda db.\, n\, d\, (\lambda z.\, b\, (a\, z)) : \text{GNP} \to \text{ADJ} \to \text{GNP} \\
\mathsf{det} &= \lambda nd.\, \lambda ea.\, n\, d\, a : \text{GNP} \to \text{DET} \to \text{GNP}
\end{aligned}$$

5 Revisiting the subject and object dependencies

The typing principle we posited in the previous section must be propagated throughout the grammar. Therefore, the type assigned to nsubj and obj should no longer be GS $\to$ NP $\to$ GS but GS $\to$ GNP $\to$ GS. Similarly, the semantic type assigned to a proper name should be GNP rather than NP. let us illustrate this with a last example. Consider the following sentence:

(7) *Michael drives a red car*



This example can be handled using the following semantic recipes, which implement the principles we have discussed in this abstract:

$$\begin{aligned}
\text{A} = \text{SOME} &= \lambda pq.\, \exists x.\, (p\, x) \wedge (q\, x) : \text{DET} \\
\text{RED} &= \lambda nx.\, (n\, x) \wedge (\mathbf{red}\, x) : \text{ADJ} \\
\text{CAR} &= \lambda da.\, d\, (a\, \mathbf{car}) : \text{GNP} \\
\text{MICHEL} &= \lambda dap.\, p\, \mathbf{m} : \text{GNP} \\
\text{DRIVE} &= \lambda p.\, \exists e.\, (\mathbf{drive}\, e) \wedge (p\, e) : \text{GS} \\
\mathsf{amod} &= \lambda na.\, \lambda db.\, n\, d\, (\lambda z.\, b\, (a\, z)) : \text{GNP} \to \text{ADJ} \to \text{GNP} \\
\mathsf{det} &= \lambda nd.\, \lambda ea.\, n\, d\, a : \text{GNP} \to \text{DET} \to \text{GNP} \\
\mathsf{nsubj} &= \lambda vn.\, \lambda p.\, n\, \text{SOME}\, (\lambda x.\, x)\, (\lambda x.\, v\, (\lambda e.\, (\mathbf{agent}\, e\, x) \wedge (p\, e))) : \text{GS} \to \text{NP} \to \text{GS} \\
\mathsf{obj} &= \lambda vn.\, \lambda p.\, n\, \text{SOME}\, (\lambda x.\, x)\, (\lambda x.\, v\, (\lambda e.\, (\mathbf{theme}\, e\, x) \wedge (p\, e))) : \text{GS} \to \text{NP} \to \text{GS}
\end{aligned}$$

The reader may then check that the four possible expressions that encode the above dependency structure yield all the same semantic interpretation[2] of sentence (7), namely:

$$\lambda f.\, \exists x.\, (\mathbf{car}\, x) \wedge (\mathbf{red}\, x) \wedge (\exists e.\, (\mathbf{drive}\, e) \wedge (\mathbf{agent}\, e\, \mathbf{m}) \wedge (\mathbf{theme}\, e\, x) \wedge (f\, e))$$

6 Conclusions

We have discussed and elaborated some principles that provide the basis for a formal theory of dependency semantics. The resulting system satisfies several interesting properties that the format of this extended abstract does not allow us to illustrate further. In particular, the toy semantic grammar that supports our last example allows incomplete dependency structures to be assigned semantic interpretations, showing therefore some robustness. It also provides an effective treatment of scope ambiguities.

References

1. L. Champollion. The interaction of compositional semantics and event semantics. *Linguistic & Philosophy*, 38(1):31–66, 2015.
2. I. Heim and A. Kratzer. *Semantics in Generative Grammar*. Blackwell Publishing, 1998.
3. Igor Mel'čuk. *Dependency Syntax: Theory and Practice*. State University of New York Press, 1988.
4. R. Montague. The proper treatment of quantification in ordinary english. In J. Hintikka, J. Moravcsik, and P. Suppes, editors, *Approaches to natural language: proceedings of the 1970 Stanford workshop on Grammar and Semantics*, Dordrecht, 1973. Reidel. Reprinted: [5, pages 247–270].
5. R. Montague. *Formal Philosophy: selected papers of Richard Montague, edited and with an introduction by Richmond Thomason*. Yale University Press, 1974.
6. J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).
7. S Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, and M. Lapata. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140, 2016.

[2] up to conjunction commutativity