Research Report AI-1994-01

An Empirically Motivated Reinterpretation of Dependency Grammar

Michael A. Covington

Artificial Intelligence Programs
The University of Georgia
Athens, Georgia 30602 U.S.A.

An Empirically Motivated Reinterpretation of Dependency Grammar

Michael A. Covington

Artificial Intelligence Programs
The University of Georgia
Athens, Georgia 30602–7415
mcovingt@ai.uga.edu

Abstract

Dependency grammar is usually interpreted as equivalent to a strict form of X–bar theory that forbids the stacking of nodes of the same bar level (e.g., $\overline{\mathrm{N}}$ immediately dominating $\overline{\mathrm{N}}$ with the same head). But adequate accounts of *one*–anaphora and of the semantics of multiple modifiers require such stacking and accordingly argue against dependency grammar. Dependency grammar can be salvaged by reinterpreting its claims about phrase structure, so that modifiers map onto binary–branching X–bar trees rather than "flat" ones.

1 Introduction

Arguments for stacked X–bar structures (such as $\overline{\mathrm{N}}$ immediately dominating $\overline{\mathrm{N}}$ with the same head) are arguments against dependency grammar as normally understood. This paper reviews the dependency grammar formalism, presents evidence that stacked $\overline{\mathrm{N}}$ structures are required, and then proposes a reinterpretation of dependency grammar to make it compatible with the evidence.

2 Dependency grammar

2.1 The formalism

Dependency grammar (DG) describes syntactic structure in terms of links between individual words rather than constituency trees. DG has its roots in Arabic and Latin traditional grammar; its modern advocates include Tesnière (1959), Robinson (1970), Starosta (1988), Mel'čuk (1987), Hudson (1980a, 1980b, 1990), and myself (Covington 1990).

The fundamental relation in DG is between head and dependent. One word (usually the main verb) is the head of the whole sentence; every other word depends on some head, and may itself be the head of any number of dependents. The rules of grammar then specify what heads can take what dependents (for example, adjectives depend on nouns, not on verbs). Practical DGs distinguish

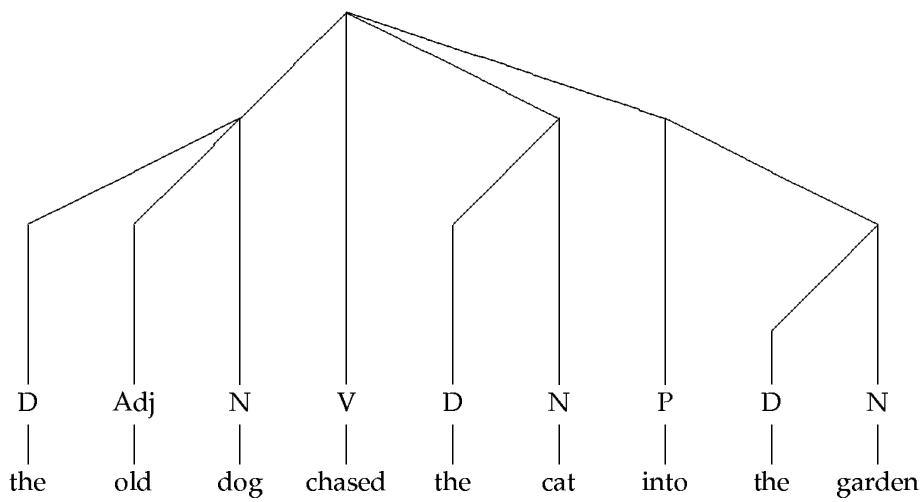


Figure 1: A dependency analysis. Heads are connected to dependents by downward–sloping lines.

various types of dependents (complement, adjunct, determiner, etc.), but the details are not important for my argument.

Figure 1 shows, in the usual notation, a dependency analysis of *The old dog chased the cat into the garden.* Here *chased* is the head of the sentence;[1] *dog* and *cat* depend on *chased*; *the* and *old* depend on *dog;* and so on.

2.2 Constituency in DG

Dependency grammar still recognizes constituents, but they are a defined rather than a basic concept. The usual definition is that a constituent consists of any word plus all its dependents, their dependents, and so on recursively. (Tesnière calls such a constituent a NŒUD.) Thus the constituents in Figure 1 are (in addition to the individual words):

the old dog (headed by *dog*)
the garden (headed by *garden*)
into the garden (headed by *into*)
the old dog chased the cat into the garden (headed by *chased*).

[1]But it would be completely compatible with the formalism to postulate that the head of the sentence is a potentially empty INFL or the like. Then, in Fig. 2, the VP would be a constituent.

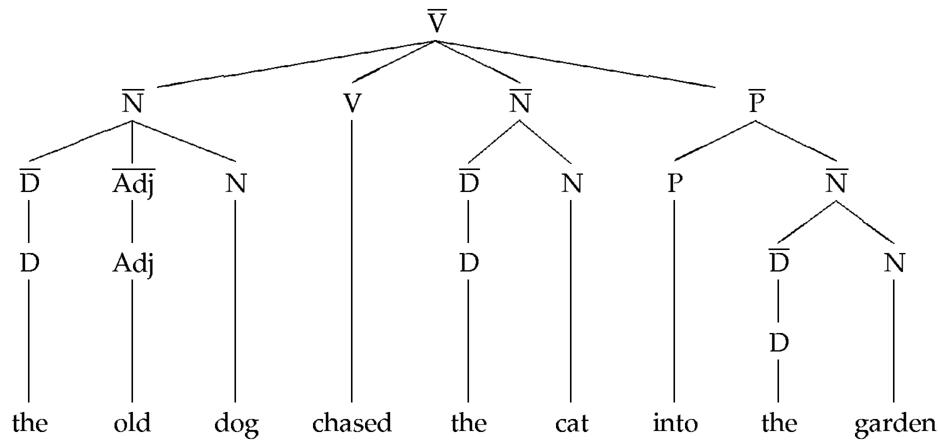


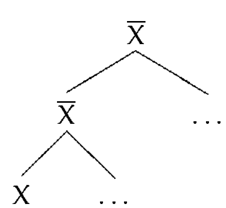
Figure 2: X–bar translation of the structure in Figure 1.

There is a rule that, at least in English, every constituent must be a contiguous string of words (Robinson 1970; Hudson 1990:114–120).

Because of its assertion that every constituent has a head, DG formalism is equivalent to a particular strict form of X–bar theory in which:

- There is only one non–terminal bar level (i.e., X and $\overline{\mathrm{X}}$, but not $\overline{\overline{\mathrm{X}}}$, $\overline{\overline{\overline{\mathrm{X}}}}$, etc.);
- Apart from bar level, X and the $\overline{\mathrm{X}}$ immediately dominating it cannot differ in any way, because they are "really" the same node;
- There is no "stacking" of $\overline{\mathrm{X}}$ nodes (an $\overline{\mathrm{X}}$ node cannot dominate another $\overline{\mathrm{X}}$ with the same head).

The third of these observations is the critical one: structures of the form



are ruled out. Figure 2 shows Figure 1 recast into X–bar theory according to this interpretation.

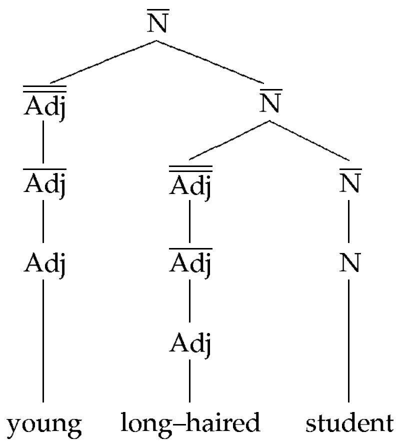


Figure 3: *One* can stand for any of the three $\overline{\mathrm{N}}$'s here.

3 Difficulty 1: The proform *one*

Dependency grammar runs into substantial difficulty trying to account for the proform *one.* The generalization to be captured is that *one* stands for a constituent larger than the N but smaller than the NP:

a young long–haired student and an older short–haired one
a young long–haired student and an older one
a young long–haired student and another one

The standard X–bar analysis (Andrews 1983, Radford 1988:189) accounts for this behavior elegantly by postulating that *one* is a pro-$\overline{\mathrm{N}}$, and that $\overline{\mathrm{N}}$'s form stacked structures (Figure 3). Dependency grammar can do no such thing, because in dependency grammar as normally understood, all the modifiers hang from the same $\overline{\mathrm{N}}$ node (Figure 4).

Further, the stacked $\overline{\mathrm{N}}$ analysis predicts a structural ambiguity if there are modifiers on both sides of the head noun — and the behavior of *one* shows that this ambiguity is real. Each $\overline{\mathrm{N}}$ in either tree in Figure 5 can be the antecedent of *one*:

the long–haired student from Cambridge and a short–haired one from Oxford
the long–haired student from Cambridge and a short–haired one
the long–haired student from Cambridge and one from Oxford
this long–haired student from Cambridge and the other one

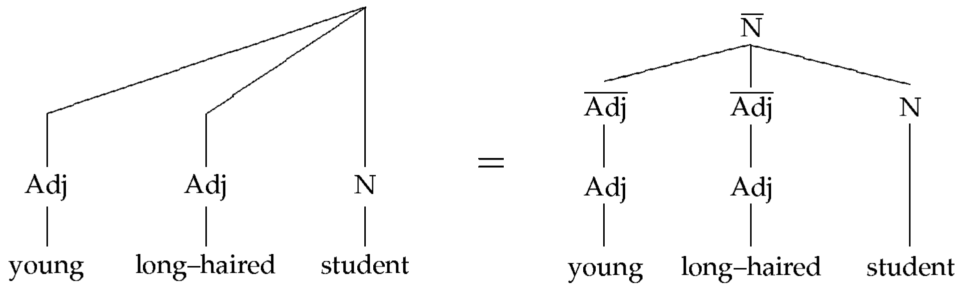


Figure 4: This dependency analysis (shown with its X–bar equivalent) lacks the stacked $\overline{\text{N}}$'s needed to account for *one*.

Again dependency grammar is left high and dry — DG formalism can recognize neither the stacking nor the ambiguity, because all the modifiers have the same head.

4 Difficulty 2: Semantics of multiple modifiers

A second difficulty with dependency grammar comes from semantics. Dahl (1980) points out that proximity to the head affects the meaning of certain modifiers. A *typical French house* is something typical of French houses, not merely a house that is French and typical. Semantically, at least, its structure is therefore:

[*typical* [*French house*]]

which is consistent with a stacked $\overline{\text{N}}$ analysis. But this grouping cannot be expressed by dependency grammar, because as far as DG is concerned, *typical* and *French* are dependents of *house*, and there is no intermediate syntactic structure.

Andrews (1983) points out that the same thing happens with verbs. Contrast:

[[*knocked twice*] *intentionally*] (acted on one intention, to knock twice)
[[*knocked intentionally*] *twice*] (had the intention two times)

These argue strongly for stacking of $\overline{\text{V}}$'s, or at least for something comparable on the semantic level.

Note by the way that if there are modifiers on both sides of the verb, an ambiguity arises just as it did with nouns: *intentionally knocked twice* is ambiguous between [[*intentionally knocked*] *twice*] and [*intentionally* [*knocked twice*]].

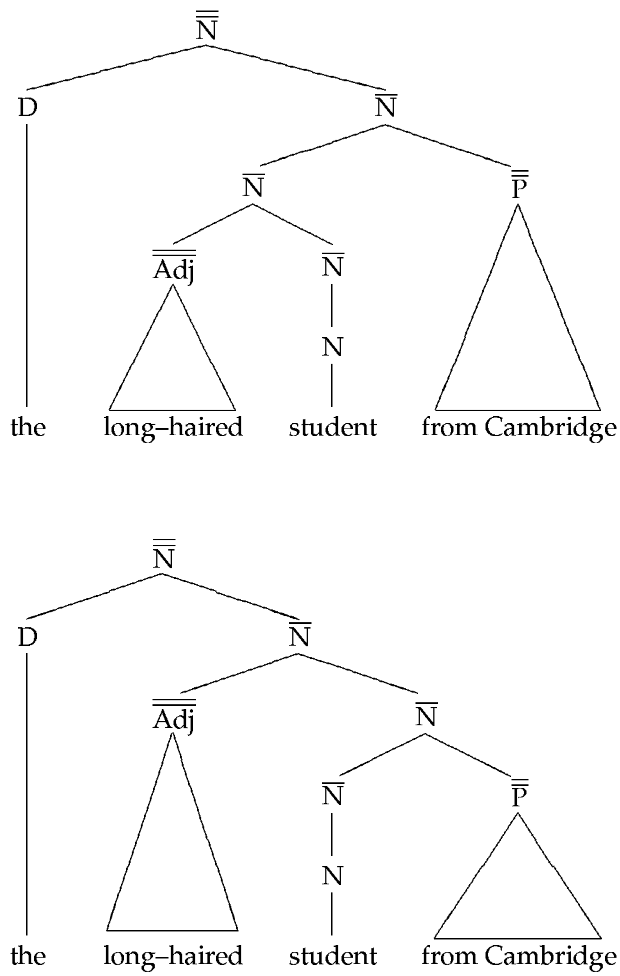


Figure 5: Dependency grammar cannot express this structural ambiguity; DG can only say that both *long–haired* and *from Cambridge* modify *student*.

Crucially, these phenomena entail that if one adopts a non–stacked syntax such as that mandated by the standard interpretation of DG, then the semantic component of the grammar must know not only the grammatical relations recognized by the syntax, but also the comparative proximity of the various modifiers to the head.

5 Reinterpreting dependency grammar

Dependency grammar can be salvaged from this mess by reinterpreting its claims about phrase structure. Recall that in a dependency grammar, constituency is a defined concept. The solution is therefore to change the definition. Specifically, instead of being considered equivalent to flat X–bar trees, dependency structures can be mapped onto X–bar trees that introduce stacking in a principled way.[2]

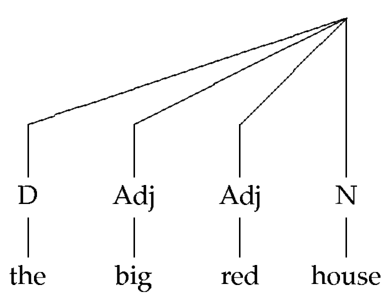
Here is a sketch of such a reinterpretation, consistent with current X–bar theory. Given a head (X) and its dependents, attach the dependents to the head by forming stacked $\overline{X}$ nodes as follows:

1. Attach subcategorized complements first, all under the same $\overline{X}$ node.[3] If there are none, create the $\overline{X}$ node anyway.

2. Then attach modifiers, one at a time, by working outward from the one nearest the head noun, and adding a stacked $\overline{X}$ node for each.

3. Finally, create an $\overline{\overline{X}}$ node at the top of the stack, and attach the specifier (determiner), if any.

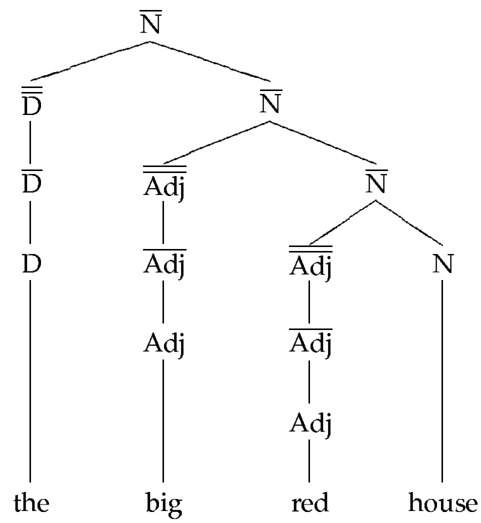
Thus the dependency structure



maps, under the new interpretation, to the stacked structure:

[2] This reinterpretation was suggested by Hudson's proposal (1980b:499–501, 1990:149–150) that the semantic effect of proximity of the head is due to a parsing effect. Since parsing is nothing if not syntactic, it seems desirable to incorporate this proposal into the syntactic theory.

[3] Actually, it is immaterial to my argument whether all the complements hang from the same node or whether they, too, are introduced by binary branching, like the adjuncts.



The distinction between specifier, modifier, and complement is already needed in dependency grammar, so this interpretation does not require anything new in the dependency formalism (Hudson 1990:202–211).

Note that if there are modifiers both before and after the head, the resulting X–bar tree is not unique — and this non–uniqueness is desirable, because the resulting alternatives, such as

[[*long–haired student*] *from Cambridge*] : [*long–haired* [*student from Cambridge*]]

[[*intentionally knocked*] *twice*] : [*intentionally* [*knocked twice*]]

are exactly the ones required by the evidence.

6 Conclusion

The alert reader may wonder, at this point, whether dependency grammar has been salvaged or rather refuted, because under the new interpretation, DG is a notational variant of current X–bar theory. To this I have several replies:

1. It should not be surprising when separate theories of the same phenomena develop convergently.

2. DG always WAS a notational variant of X–bar theory; I have merely brought its implicit X–bar theory up to date.

3. DG still imposes stricter requirements than transformational grammar, because in DG, violations of X–bar theory are flatly impossible, not just undesirable.

In any case, the dependency perspective on sentence structure has proved its worth not only in syntactic theorizing, but also in language teaching, parsing, and other practical applications. Indeed, dependency concepts, such as government and c–command, are becoming increasingly prominent in transformational grammar. Dependency grammar can complement other approaches to syntax in much the same way that relational grammar, fifteen years ago, provided an organizing perspective on what had previously been a heterogeneous set of syntactic transformations.

References

Andrews, Avery, III (1983) A note on the constituent structure of modifiers. *Linguistic Inquiry* 14:695–697.

Covington, Michael A. (1990) Parsing discontinuous constituents in dependency grammar. *Computational Linguistics* 16:234–236.

Dahl, Östen (1980) Some arguments for higher nodes in syntax: a reply to Hudson's 'Constituency and dependency.' *Linguistics* 18:485–488.

Hudson, Richard (1980a) Constituency and dependency. *Linguistics* 18:179–198.

Hudson, Richard (1980b) A second attack on constituency: a reply to Dahl. *Linguistics* 18:489–504.

Hudson, Richard (1990) *English word grammar.* Oxford: Basil Blackwell.

Mel'čuk, Igor A. (1987) *Dependency syntax: theory and practice.* Albany: State University of New York Press.

Radford, Andrew (1988) *Transformational grammar.* Cambridge: Cambridge University Press.

Robinson, Jane J. (1970) Dependency structures and transformational rules. *Language* 46:259–285.

Starosta, Stanley (1988) *The case for lexicase.* London: Pinter.

Tesnière, Lucien (1959) *Éléments de syntaxe structurale.* Paris: Klincksieck.