

Examples of Expanded Categorical Syntax

Gary Hardegree

Department of Philosophy
University of Massachusetts
Amherst, MA 01003

1.	Introduction.....	2
2.	Simple Examples	2
3.	Re-categorizing Common-Noun Phrases.....	4
4.	Re-categorizing Bare-Adjectives	5
5.	(The Prospect of) Re-Categorizing Copular-Be	6
6.	Re-Locating ‘not’ in its Natural Position.....	8
7.	Passives	9
8.	Function-Like Nouns	12
1.	Genitive Nouns	12
2.	Initial Analysis	12
3.	Examples.....	13
9.	Type-Logical Accounts of Well-Known Type-Shifting Techniques.....	15

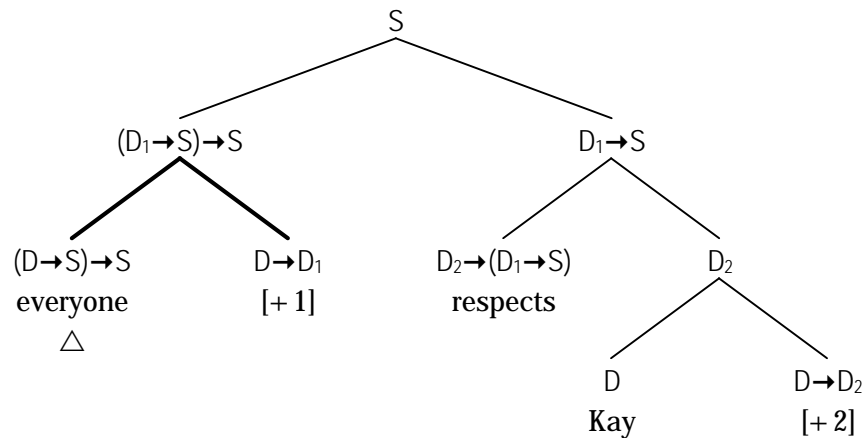
1. Introduction

In the current chapter, we illustrate the techniques of Expanded Categorical Syntax by analyzing several examples. Along the way, we propose adjustments to our theory. In particular, we propose to re-categorize bare-adjectives, common-noun phrases, and copular-be. We also examine passives and function-like nouns.

We begin with a few simple examples.

2. Simple Examples

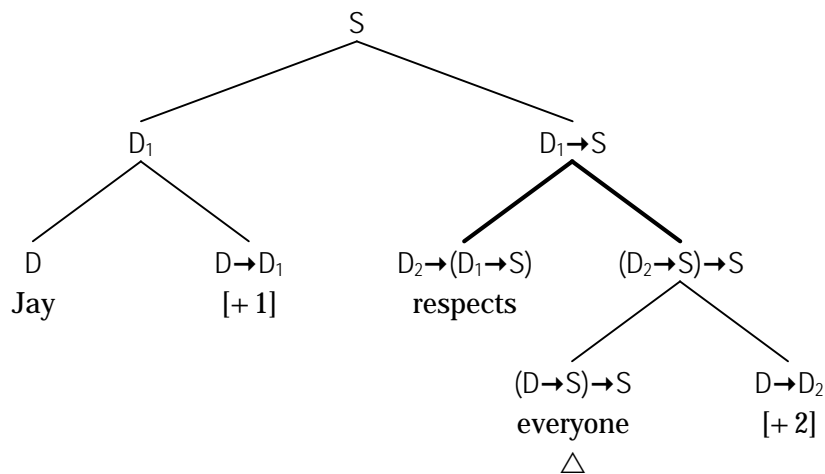
1. everyone respects Kay



All the compositions in the above tree are accomplished by way of *modus ponens*, except the highlighted one, which is accomplished using the inference principle we call *inflection*,¹ given as follows.

$$(A \rightarrow C) \rightarrow D ; A \rightarrow B \vdash (B \rightarrow C) \rightarrow D \quad \text{[inflection]}$$

2. Jay respects everyone

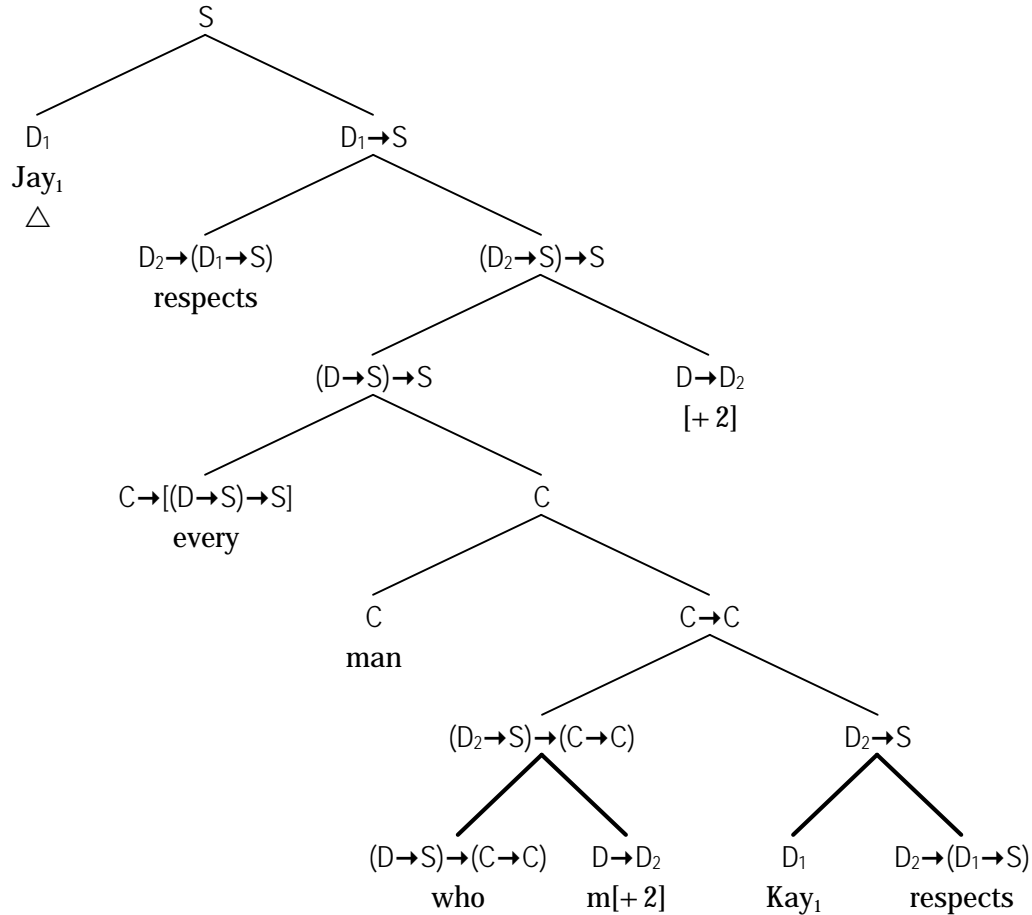


¹ This is precisely the reason we call it *inflection*. There is no precedent in logic for this nomenclature.

First, ‘everyone₂’ is composed using *inflection*, as in the previous example. This phrase then combines with ‘respects’ to produce a VP [D₁→S] using an inference principle that combines permutation and transitivity, and is given as follows.

$$A \rightarrow (B \rightarrow C) ; (A \rightarrow C) \rightarrow D \vdash B \rightarrow D \quad [\text{permutation} + \text{transitivity}]$$

3. Jay respects every man whom Kay respects



First, notice that ‘Kay₁’ has nominative-inflection, which means that it does not serve as the object of ‘respects’. Nevertheless, it combines with ‘respects’ to form an accusative-predicate, by which we mean a phrase of type D₂→S. The composition is accomplished using *secondary modus ponens*, which is given as follows.

$$A \rightarrow (B \rightarrow C) ; B \vdash A \rightarrow C \quad [\text{secondary modus ponens}]$$

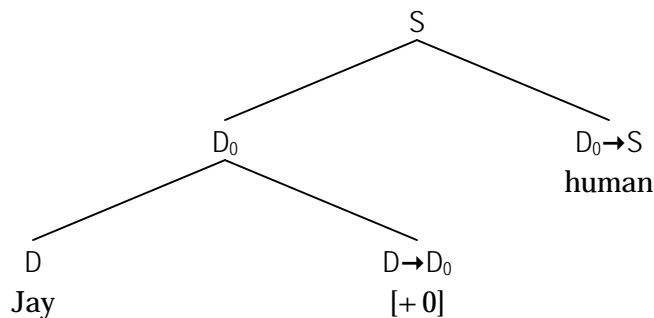
Next, notice that ‘whom’ is accusative-inflected, where the relevant composition-principle is, once again, *inflection*. The resulting phrase is a functor that takes an accusative-predicate [D₂→S] as input and yields an adjective [C→C] as output. Notice, in this connection, that the phrase ‘Kay respects’ is an accusative-predicate, and is accordingly appropriate input for this functor.

3. Re-categorizing Common-Noun Phrases

As currently formulated, our grammatical theory treats common-noun phrases as a primitive syntactic type C . As it turns out, there are numerous reasons to categorize common-noun phrases so they are "isomorphic" to, but not identical to, VPs. In our revised grammatical theory, we can make these subtle distinctions. In particular, whereas VPs are nominative predicates $[D_1 \rightarrow S]$, we propose to treat CNPs as "nullative" predicates, as follows.

$$C \quad =_{df} \quad D_0 \rightarrow S$$

Here, 0 is the special "nullative" case-marker, which we propose to add to the usual list of case-markers. We further propose that the inflectional morpheme $[+0]$ is syntactically disallowed, at least in standard English, so that the following tree is inadmissible.²



Re-categorizing CNPs has immediate repercussions throughout the system, which are summarized in the following table.

category	old type	new type
quantifier	$CNP \rightarrow (VP \rightarrow S)$	$(D_0 \rightarrow S) \rightarrow [(D \rightarrow S) \rightarrow S]$
	$C \rightarrow [(D \rightarrow S) \rightarrow S]$	
adjective	$CNP \rightarrow CNP$	$(D_0 \rightarrow S) \rightarrow (D_0 \rightarrow S)$
	$C \rightarrow C$	
relative-clause functor	$VP \rightarrow Adj$	$(D \rightarrow S) \rightarrow [(D_0 \rightarrow S) \rightarrow (D_0 \rightarrow S)]$
	$(D \rightarrow S) \rightarrow (C \rightarrow C)$	

Note that we will continue to use 'C' as shorthand for 'D₀→S'. We will expand the definition basically only when it is needed to account for a particular type-logical composition.

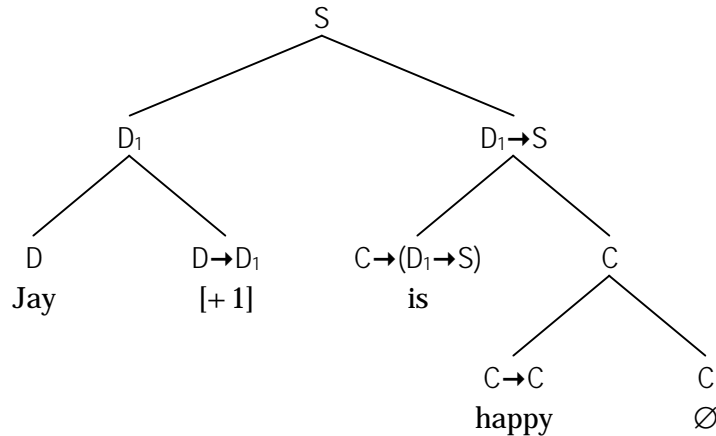
² This is not to say that 'Jay human' is inadmissible *per se*; it's just not a sentence. A common noun can be used appositionally, as in:

Austin Powers: international man of mystery

Admittedly 'Jay: human' is not as catchy a movie title, but the two expressions probably have the same overall syntactic form. Later, we discuss exactly what type these expressions (may) have.

4. Re-categorizing Bare-Adjectives

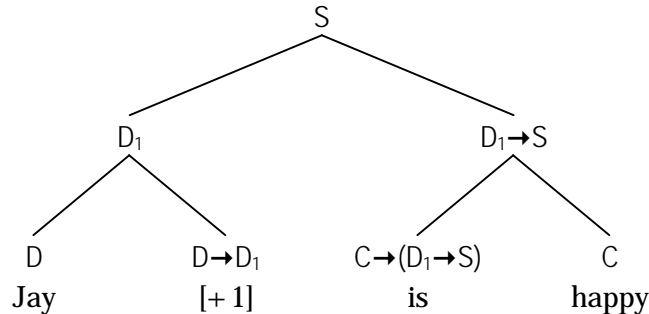
Currently, we treat adjectives as CNP-modifiers, and we treat bare-adjectives as having understood arguments, as in the following tree, where $C \stackrel{\text{df}}{=} D_0 \rightarrow S$.



We now propose a revised account according to which there is a privileged class of *bare-adjectives*, which we type-render just like CNPs, as follows.³

$$\begin{aligned} \text{type}(\text{bare-adjective}) &= C \\ &\stackrel{\text{df}}{=} D_0 \rightarrow S \end{aligned}$$

In that case, the above tree may be simplified as follows.



We further propose a (covert) morpheme – *mod* – which converts bare-adjectives and CNPs into modifier-adjectives, as follows.

$$\begin{aligned} \text{type}(\text{mod}) &\stackrel{\text{df}}{=} C \rightarrow (C \rightarrow C) \\ &\stackrel{\text{df}}{=} (D_0 \rightarrow S) \rightarrow [(D_0 \rightarrow S) \rightarrow (D_0 \rightarrow S)] \end{aligned}$$

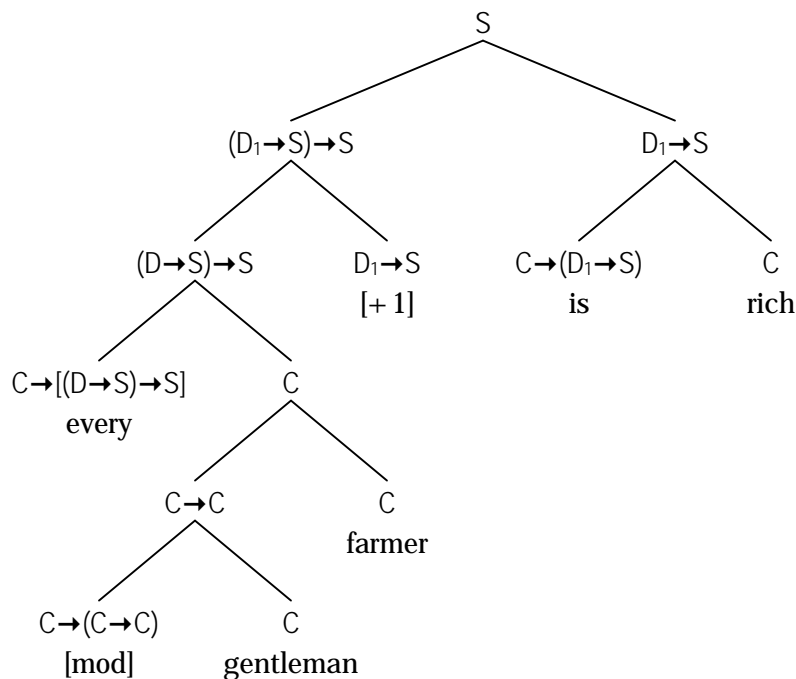
Usually, we take *mod* for granted, and simply treat every bare-adjective as automatically also a

³ We quickly note that the syntactic module is still required to tell us that bare adjectives, not to mention ‘a’-headed phrases, are not syntactically-admissible input for quantifiers, even though they are type-identical to common-noun phrases, which are suitable input. For example, the following are not syntactically well-formed, although they are not incomprehensible.

- ⊗ every tall is happy
- ⊗ every a woman is happy

If we insist on keeping these matters straight, we can instead propose various allomorphs, $N_0 \rightarrow S$, $N_{00} \rightarrow S$, $N_{000} \rightarrow S$, which behave similarly semantically, but which behave differently syntactically. The reader is free to consider such an account.

modifier-adjective. Other times, *mod* must be explicitly included – for example, when it applies to a common-noun, as in the following example.



As we later see, when we do semantics, the *mod* operator is equivalent to ‘who is (a)’, so the expression ‘gentleman farmer’ is equivalent to ‘farmer who is a gentleman’.

Not all common nouns compose this way, however. For example a potato farmer is not a farmer who is a potato! The latter construction requires a more advanced theory of adjective-modification than we are currently considering.⁴ For the moment at least, we only consider modifier-adjectives that derive from bare-adjectives. This rules out numerous CNP-modifiers, including ‘former’, ‘alleged’, and ‘imitation’. For example, a former student is not a student who is former, since the latter makes no sense.

5. (The Prospect of) Re-Categorizing Copular-Be

The original proposal concerning copular-be [the ‘is’ of predication] is the following.

$$\begin{aligned} \text{type(cop-be)} &= \text{CNP} \rightarrow \text{VP} \\ &=_{\text{df}} \text{C} \rightarrow (\text{D} \rightarrow \text{S}) \end{aligned}$$

This analysis must now be adjusted to take into account that VPs are inflected, and to take into account our re-categorization of *C*, which yields the following adjusted type-assignment.

$$\begin{aligned} \text{type(cop-be)} &= \text{CNP} \rightarrow \text{VP} \\ &= \text{C} \rightarrow (\text{D}_1 \rightarrow \text{S}) \\ &=_{\text{df}} (\text{D}_0 \rightarrow \text{S}) \rightarrow (\text{D}_1 \rightarrow \text{S}) \end{aligned}$$

⁴ The basic idea is that *some* common-nouns subcategorize for direct objects, which often are implicit, but which can be supplied by common-noun phrases. The semantic structure of ‘farmer’ then is something like “person who farms ...”, where ‘farm’ is a transitive verb, so ‘potato farmer’ then means “person who farms potatoes” according to this reading.

We *could* go one step further, offering a logically stronger and simpler account of copular-be as follows.

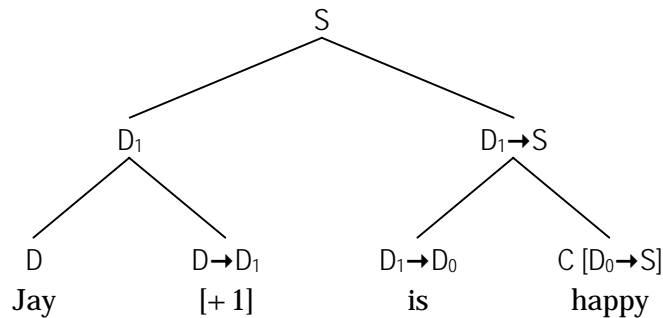
$$\text{type}(\text{cop-be}) = D_1 \rightarrow D_0$$

In this connection, note in particular the following inference principle.

$$D_1 \rightarrow D_0 \vdash (D_0 \rightarrow S) \rightarrow (D_1 \rightarrow S)$$

The following is an example, using the bare-adjective ‘happy’.

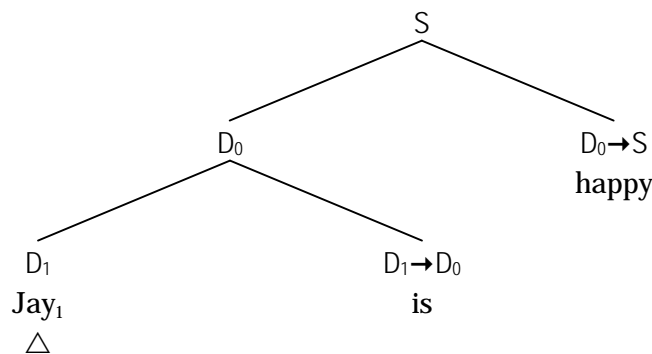
Jay is happy



Notice that the VP ‘is happy’ is composed from ‘is’ and ‘happy’ using the principle of transitivity, which is given as follows.

$$B \rightarrow C ; A \rightarrow B \vdash A \rightarrow C \quad \text{[transitivity]}$$

Although this is an inviting prospect, it *suggests* that the following analysis of ‘Jay is happy’ is *preferable* to the previous analysis.



Here, by ‘preferable’, we mean “computationally more efficient”. Notice, in particular, that the derivations in the second tree involve only *modus ponens*, whereas the derivations in the former involve transitivity as well.

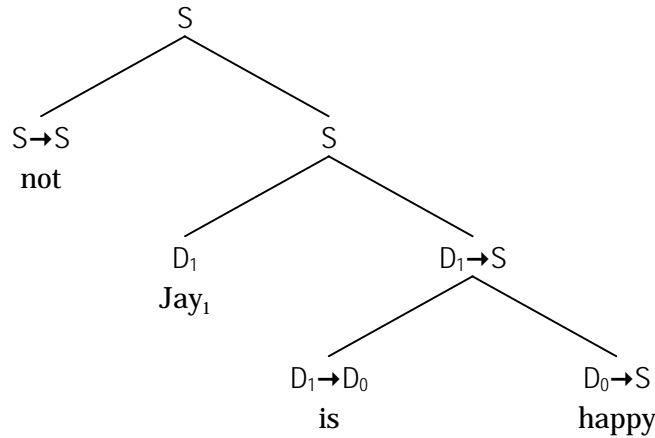
For this reason, we will maintain our account of copular-be as the following.

$$\begin{aligned} \text{type}(\text{cop-be}) &= C \rightarrow (D_1 \rightarrow S) \\ &=_{\text{df}} (D_0 \rightarrow S) \rightarrow (D_1 \rightarrow S) \end{aligned}$$

6. Re-Locating ‘not’ in its Natural Position

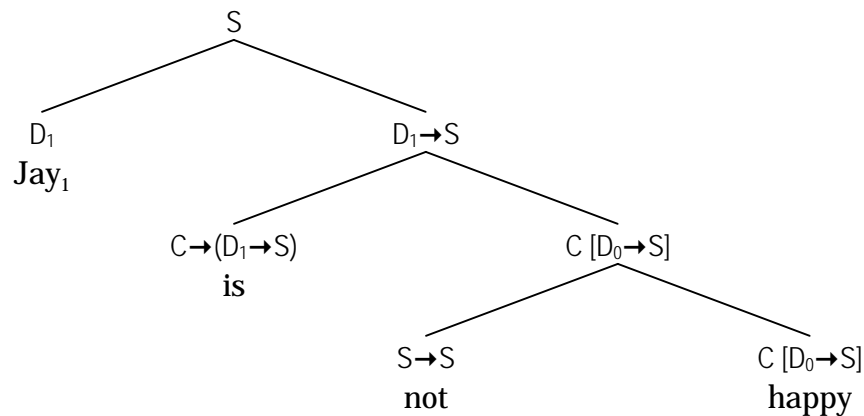
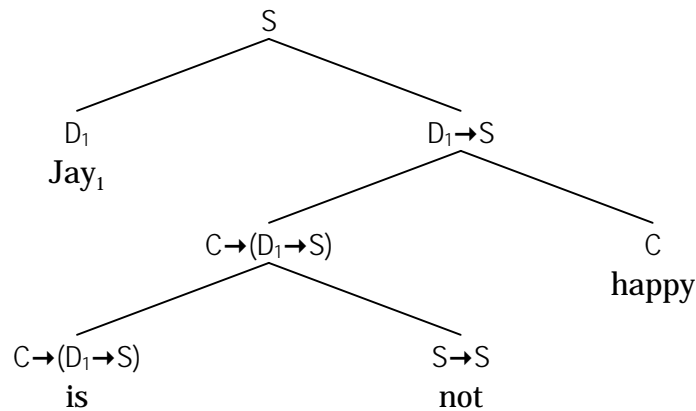
We officially treat ‘not’ as a functor of type $S \rightarrow S$. Using standard categorial syntax, in which only functor-application is allowed, the following is an example analysis.

1. Jay is not happy



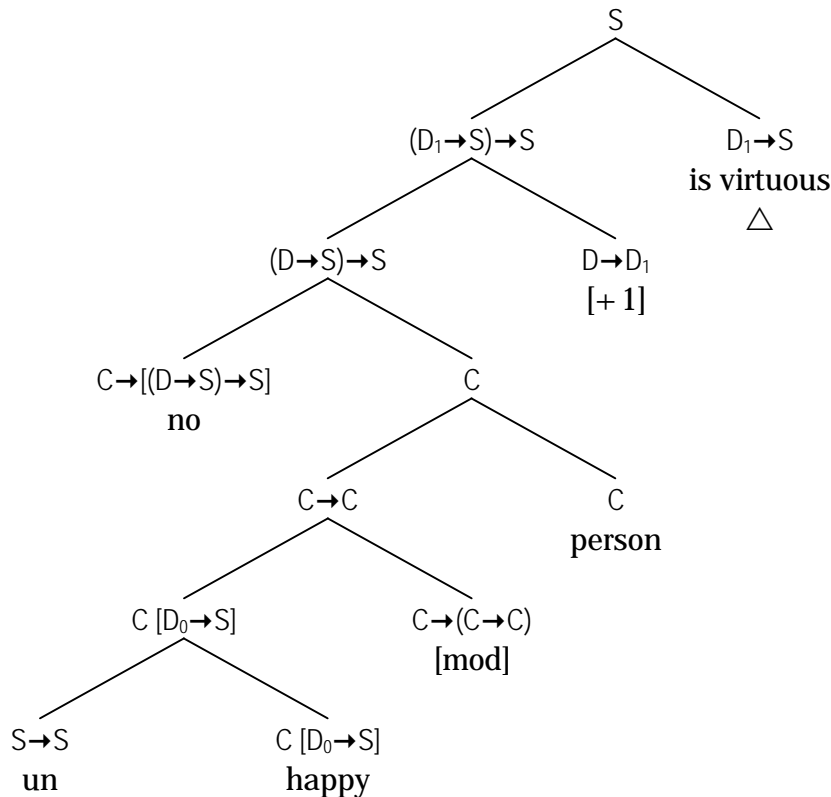
The problem is that ‘not’ is misplaced. In order to maintain the above syntactic-analysis, we must additionally postulate a syntactic/phonetic transformation that moves ‘not’ down into its "natural" position.

By contrast, if we use revised categorial syntax, which licenses a much wider array of admissible compositions, we can analyze the above sentence in either of the following ways.



The first one is more plausible in consideration of the contraction ‘isn’t’. The latter is more plausible in consideration of the contraction ‘unhappy’. The latter construction is further illustrated in the following example.

2. no unhappy person is virtuous



Note that the *mod* functor is applied to ‘unhappy’, not to ‘happy’. This is not required syntactically, since ‘not’ just as easily modifies a modifier-adjective, but it is required semantically.⁵

7. Passives

We next tackle the matter of passive verb forms. The distinction between active-voice and passive-voice is illustrated in the following.

Jay respects Kay	[active]
Kay is respected by Jay	[passive]

Rather than follow a transformational route, according to which the latter is a surface form obtained from the former underlying form by a separately hypothesized active-passive transformation, we propose to treat ‘ed’ and ‘by’ as grammatically autonomous morphemes.

In particular, we propose the following type-analyses.

$$\text{type(passive-ed)}^6 = [D_2 \rightarrow (D_1 \rightarrow S)] \rightarrow [D_5 \rightarrow C]$$

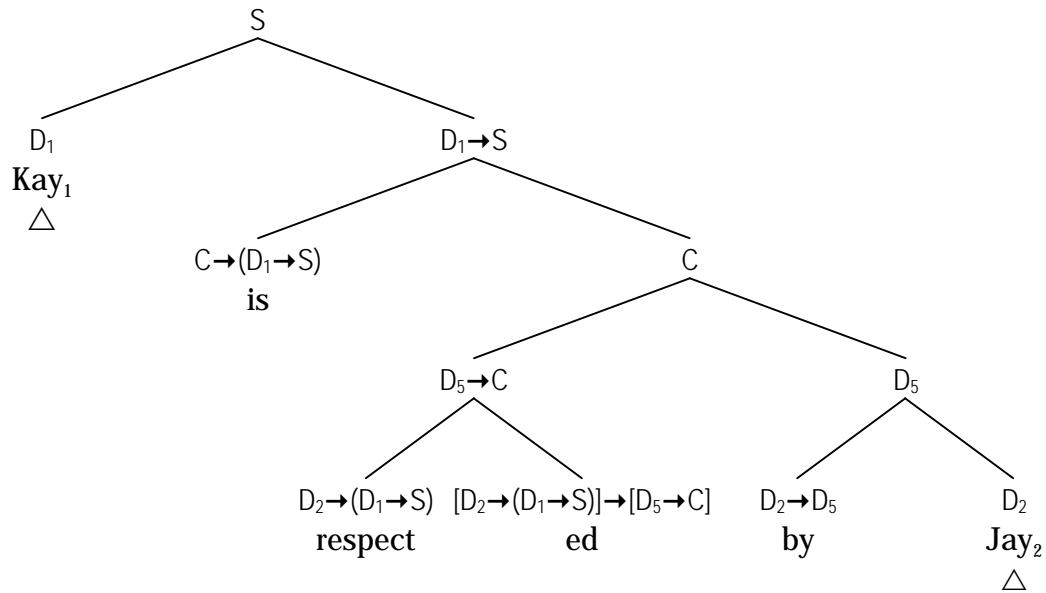
⁵ In particular, if we do it the other way around, ‘unhappy person’ ends up meaning “thing that is not a happy person”, rather than “person who is not happy”.

⁶ The suffix ‘ed’ is also used in English to produce *past tense* and *perfect aspect* [the latter is also achieved by ‘en’ for many irregular verbs]. Past tense and perfect aspect are subtly inter-connected, and often conflated in ordinary grammar.

$$\text{type}(\text{by}) = D_2 \rightarrow D_5$$

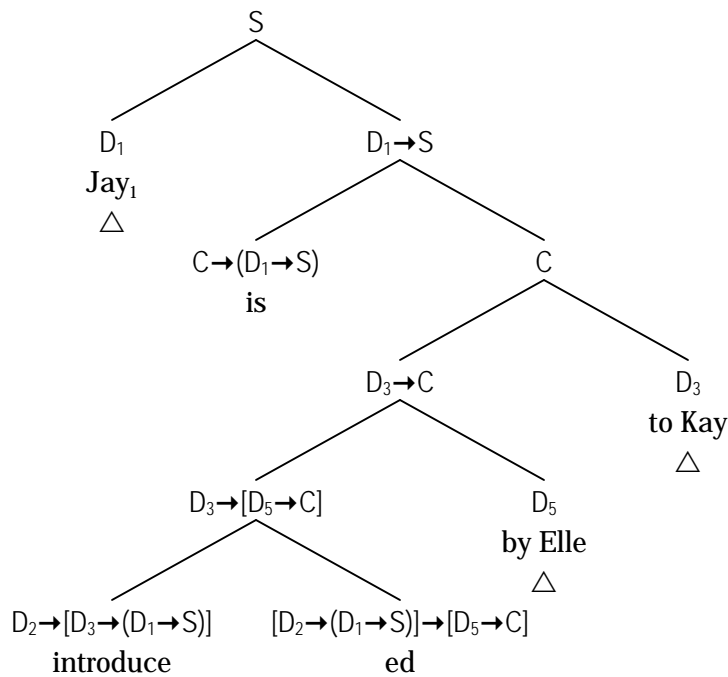
This is illustrated in the following examples.

1. Kay is respected by Jay

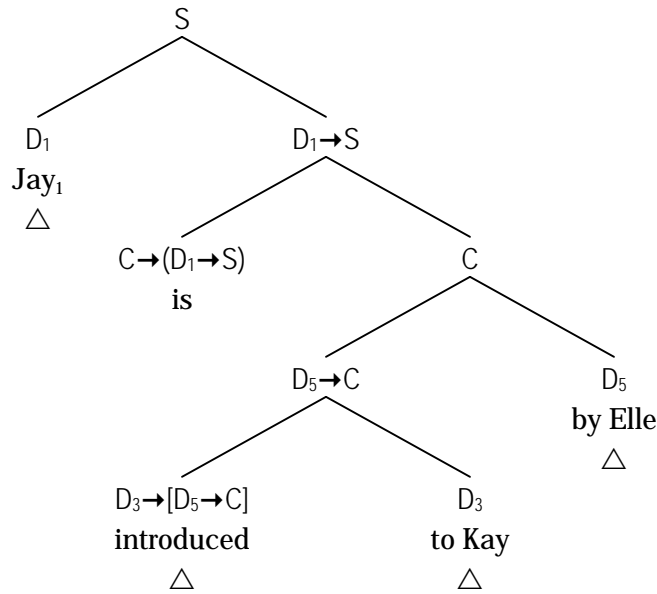


Note that the phrase 'respected by Jay' is a *bare adjective*.

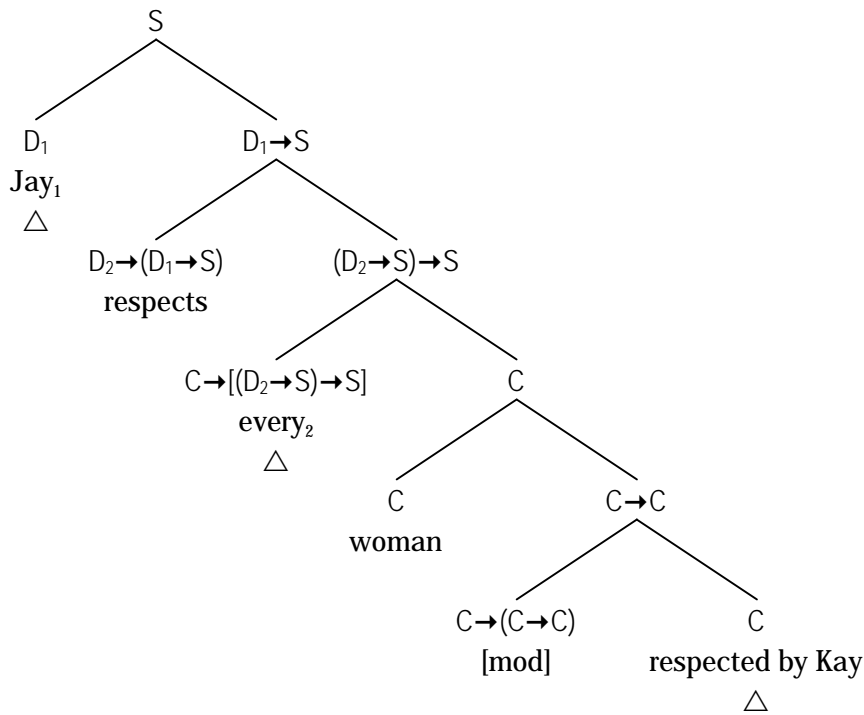
2. Jay is introduced by Elle to Kay



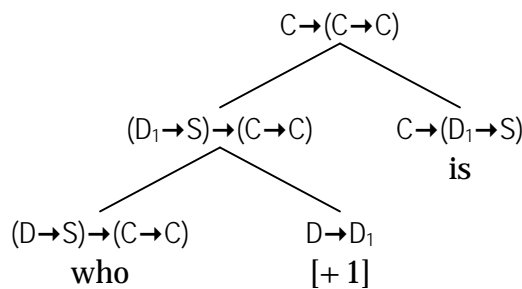
3. Jay is introduced to Kay by Elle



4. Jay respects every woman respected by Kay



Notice that the bare-adjective ‘respected by Kay’ is converted into a modifier-adjective by prefixing the mod-operator. Notice that the mod-operator is type-logically equivalent to ‘who is’, as seen in the following derivation, which employs inflection and then transitivity.



8. Function-Like Nouns

1. Genitive Nouns

Common-nouns divide into two broad classes.

- (1) ordinary common-nouns
- (2) relational common-nouns

Whereas an ordinary common-noun denotes a one-place property, a relational common-noun denotes a two-place relation, on the class of entities. Among relational common-nouns are a special subclass we call *genitive common-nouns*, or simply *genitive nouns*. Such nouns are characterized by the word ‘of’. For example, being a *mother* is tantamount to being the/a mother *of* someone; similarly, being a *brother* is tantamount to being the/a brother *of* someone. Other examples of genitive nouns include ‘friend’, ‘capital’, and ‘member’. Examples of relational nouns that are not genitive include ‘hole’ (in), ‘solution’ (to), ‘reason’ (for), and ‘passenger’ (in?).

By contrast, ordinary common-nouns are not characterized by ‘of’, or any other relational preposition. For example, being a *dog* is *not* tantamount to being the/a dog *of* (or in, or to, or for) someone or something. Most common-nouns are like ‘dog’ and not like ‘mother’; most common-nouns are ordinary.

Among genitive nouns are a special sub-class of **function-like** nouns, which include

mother, father, capital

but exclude

brother, sister, friend,

among others. What makes a relational noun function-like is not syntactic, but semantic. For example, a person has *exactly one* mother, and *exactly one* father, and a state has *exactly one* capital. On the other hand, a person can have any number of sisters, brothers, aunts, uncles, friends, etc. Mathematically speaking, the pairing of persons with their mothers is a *function*; thus the term ‘function-like’.

2. Initial Analysis

As a first approximation, we propose to categorially render function-like nouns as follows.⁷

$$D_6 \rightarrow D$$

Here, 6 encodes the *genitive-case*, which in its regular form in English is implemented by the apostrophe-s [’s] construction. The following are examples, beginning with the non-regular ones.⁸

my , your , his , her , its , our , their
whose , Jay’s , Kay’s , everyone’s , every man’s

⁷ See the chapter “Relational Nouns and Prepositions” for a more detailed account of function-like nouns.

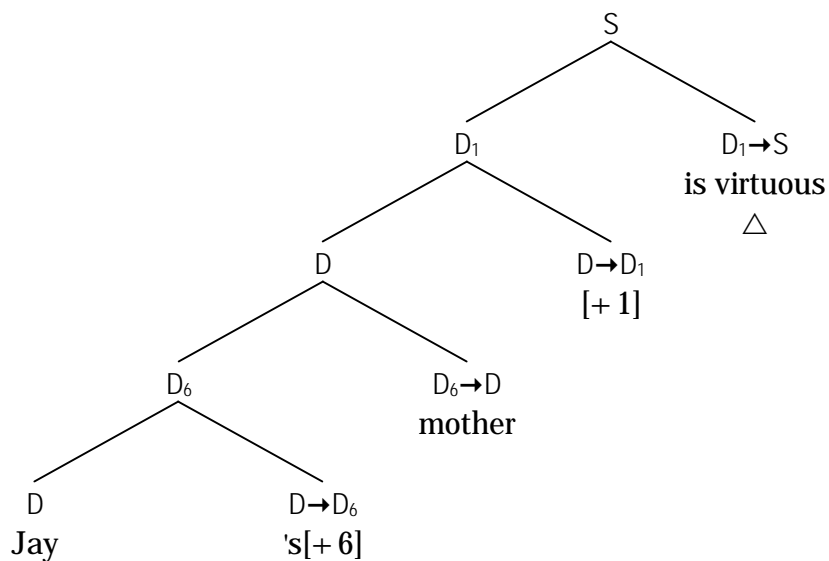
⁸ There is a secondary genitive form used in connection with ‘of’, as in the following examples.

a friend of { mine | yours | hers | ours | theirs }

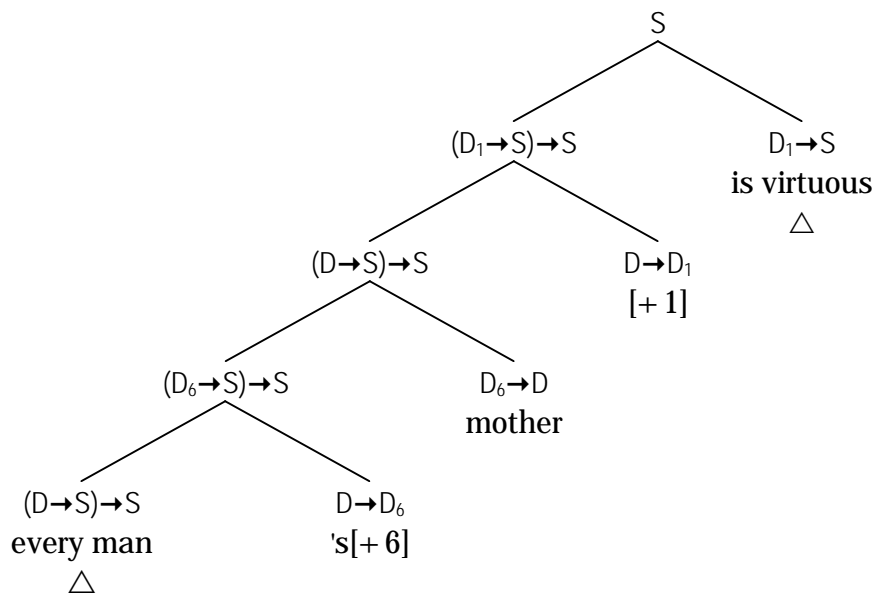
The current analysis does not accommodate the ‘of’ variant of the genitive-construction. See the chapter “Relational Nouns and Prepositions” for a more detailed discussion.

3. Examples

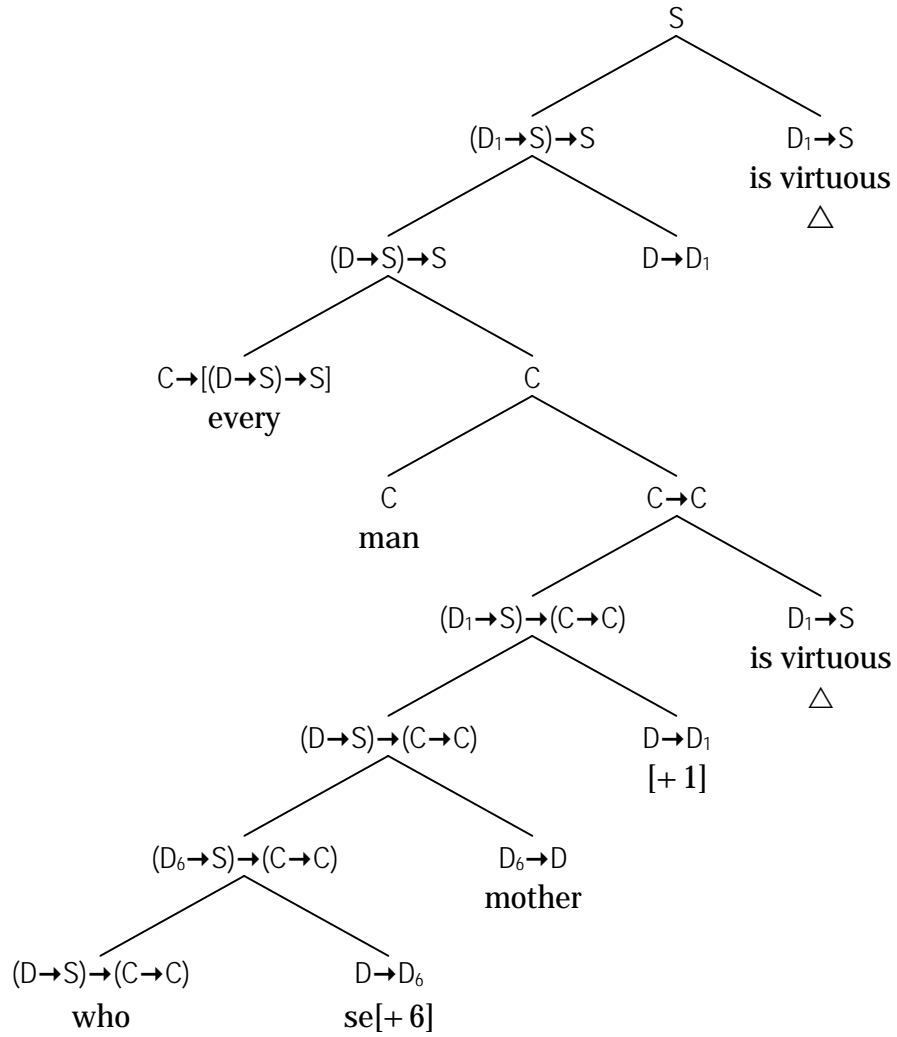
1. Jay's mother is virtuous



2. every man's mother is virtuous



3. every man whose mother is virtuous is virtuous



9. Type-Logical Accounts of Well-Known Type-Shifting Techniques

In this, the final, section of the chapter, we examine three well-known type-shifting techniques, showing how they are consequences of our general principles of grammatical composition.

1. Schönfinkel's Transform (residuation)

$$(A \times B) \rightarrow C \dashv\vdash A \rightarrow (B \rightarrow C)$$

This logical equivalence translates into categorial-logic as the equivalence of a two-place functor and its associated Schönfinkel form.⁹ For example, the following are type-logically equivalent.

$$\frac{D_2 \rightarrow (D_1 \rightarrow S)}{\frac{\quad}{(D_1 \times D_2) \rightarrow S}} \quad \begin{array}{l} \text{transitive verb} \\ \\ \text{two-place predicate} \end{array}$$

2. Montague's Transform (Lifting)

$$A \vdash (A \rightarrow B) \rightarrow B$$

This single-premise rule is the logical underpinning of the following type-shifting principle.

$$\frac{D}{(D \rightarrow S) \rightarrow S} \quad \begin{array}{l} \text{definite-noun phrase} \\ \\ \text{quantifier phrase} \end{array}$$

In other words, every definite-noun phrase automatically gives rise to an associated QP. This is the mother of all type-shifting principles, which was first introduced by Montague, who proposed that we treat all NPs as second-order predicates (QPs).¹⁰ Notice that the converse argument is not valid, which means that we are not free to convert a QP into a definite-noun phrase.

3. Generalized-Conjunction (Partee and Rooth)

Partee and Rooth¹¹ propose that ‘and’ has a multiple-type that allows it to combine any two functors of the same type, so long as that type ends in S.¹² From the viewpoint of revised categorial grammar, this amounts to proposing the following composition principle.¹³

$$\frac{\mathcal{A}_1 \rightarrow \dots \rightarrow \mathcal{A}_k \rightarrow S}{\mathcal{A}_1 \rightarrow \dots \rightarrow \mathcal{A}_k \rightarrow S}$$

⁹ Heim and Kratzer (*Semantics in Generative Grammar*) propose the term ‘Schönfinkelization’ for this type-shifting technique, which traces to Moses Schönfinkel’s “Über die Bausteine der mathematischen Logik” [Math. Ann. 92 (1924), 305-316]. Schönfinkel is known in some circles as the “father of the combinator” (<http://www.cis.upenn.edu/~steedman/moses.html>).

¹⁰ Richard Montague. “The proper treatment of quantification in ordinary English”, In J. Hintikka et al., editors, *Approaches to Natural Language*, pages 221-242. Reidel, 1973. Reprinted in R. Thomason, editor, *Formal Philosophy*. Yale University Press, Yale, 1974.

¹¹ Barbara Partee and Mats Rooth. 1983. “Generalized Conjunction and Type Ambiguity”. In R. Bauerle, C. Schwarze, and A. von Stechow (eds.), *Meaning, Use, and Interpretation of Language*, 361-366. Berlin: Walter de Gruyter.

¹² Note that linguists use the term ‘conjunction’ more broadly than logicians, so generalized-conjunction applies to all two-place connectives, not just logical-and.

¹³ Here, the parentheses are resupplied in a natural way; in particular, $A \rightarrow B \rightarrow C =_{df} A \rightarrow (B \rightarrow C)$.

$$\frac{(S \times S) \rightarrow S}{\mathcal{A}_1 \rightarrow \dots \rightarrow \mathcal{A}_k \rightarrow S}$$

Given Schönfinkel's Law, this is equivalent to the following.

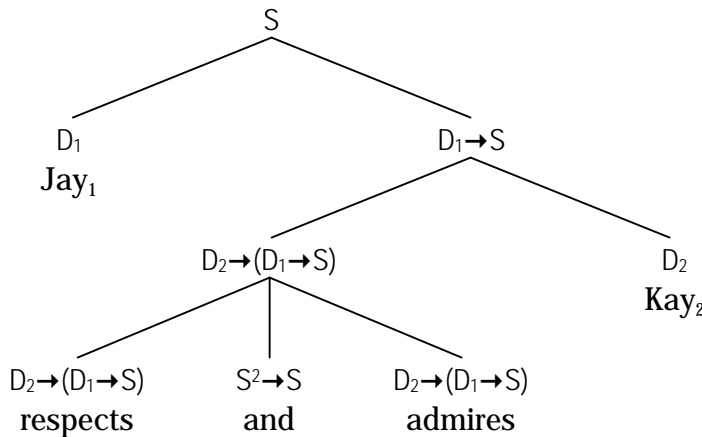
$$\frac{\begin{array}{l} (\mathcal{A}_1 \times \dots \times \mathcal{A}_k) \rightarrow S \\ (\mathcal{A}_1 \times \dots \times \mathcal{A}_k) \rightarrow S \\ (S \times S) \rightarrow S \end{array}}{(\mathcal{A}_1 \times \dots \times \mathcal{A}_k) \rightarrow S}$$

The latter, in turn, is a special case of the inference principle we propose to call *generalized-conjunction*, which is given as follows.¹⁴

$$A \rightarrow B ; A \rightarrow C ; (B \times C) \rightarrow D \vdash A \rightarrow D \quad \text{[generalized-conjunction]}$$

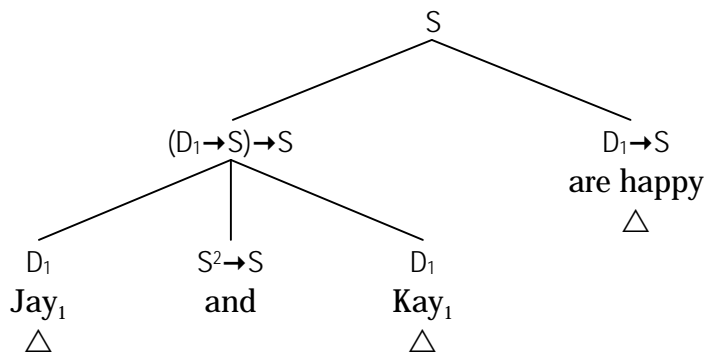
The following are some examples that employ generalized-conjunction. Recall that $S^2 =_{\text{df}} S \times S$

1. Jay respects and admires Kay.



In other words, ‘respects and admires’ is a transitive verb. Notice that we have reverted to the logician's syntactic analysis of ‘and’, according to which it has type $S^2 \rightarrow S$, rather than its Schönfinkel form $S \rightarrow (S \rightarrow S)$.

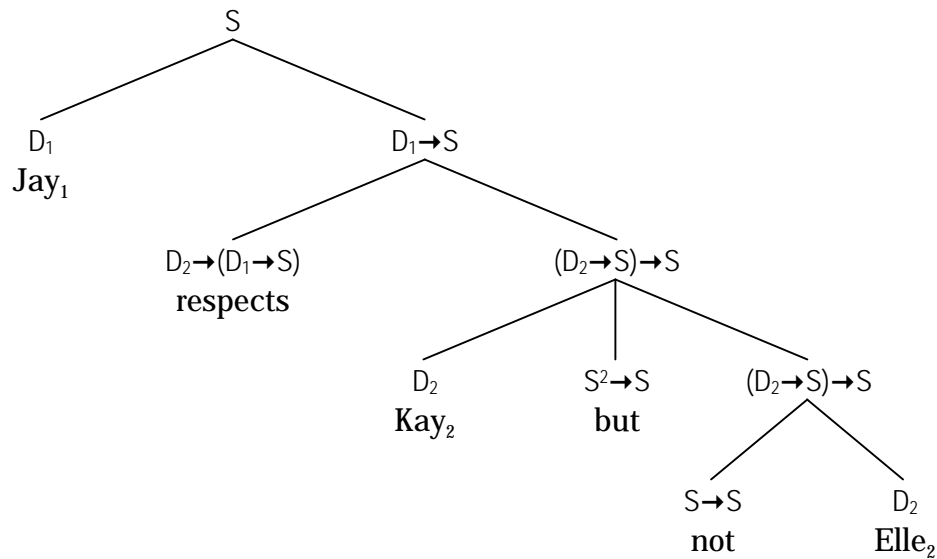
2. Jay and Kay are happy.



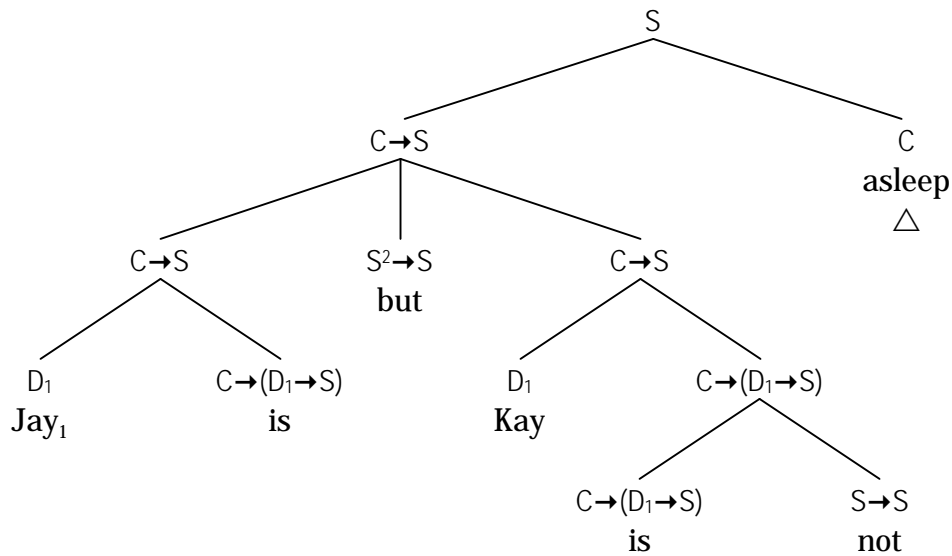
¹⁴ Note carefully that generalized-conjunction is valid in System R (Relevance Logic), but not System L (Linear Logic), which is basically why our proposed categorial logic (see chapter ‘Categorial Logic’) lies between R and L.

This is a more subtle application of generalized-conjunction, which detours through Montague-lifting, a categorial-logic principle according to which D_θ freely gives rise to $(D_\theta \rightarrow S) \rightarrow S$.¹⁵ The following two examples similarly detour through Montague-lifting.

3. Jay respects Kay but not Elle



4. Jay is, but Kay is not, asleep



¹⁵ The "logical" use of 'and' must be distinguished from the "mereological" use of 'and', which occurs in connection with irreducibly-plural predicates, as in 'Jay and Kay respect each other'. In the latter example, 'Jay and Kay' is not a QP (except trivially) but a plural-definite-noun, which refers to the plural entity Jay+Kay.