

1. Categorical Logic

Categorical Logic (CGL) is the logical calculus that governs the composition of grammatical-types. We *hypothesize* that CGL is a logical system lying between Linear Logic without Identity¹ and Relevance Logic, an intermediate system that in particular includes a means of capturing *generalized-conjunction*.

1. Examples of Entailments in CGL

- | | | |
|-----|--|----------------------------------|
| (1) | $B \rightarrow C ; A \rightarrow B \vdash A \rightarrow C$ | [Transitivity] |
| (2) | $A \rightarrow (B \rightarrow C) \vdash B \rightarrow (A \rightarrow C)$ | [Permutation] |
| (3) | $A \rightarrow (B \rightarrow C) ; B \vdash A \rightarrow C$ | [Secondary <i>Modus Ponens</i>] |
| (4) | $A \vdash (A \rightarrow B) \rightarrow B$ | [Lifting] |
| (5) | $(A \rightarrow C) \rightarrow D ; A \rightarrow B \vdash (B \rightarrow C) \rightarrow D$ | [Inflection] |
| (6) | $(A \rightarrow C) \rightarrow D ; A \rightarrow (B \rightarrow C) \vdash B \rightarrow D$ | [Permutation + Transitivity] |
| (7) | $B \rightarrow C ; A \times B \vdash A \times C$ | [Addition] |
| (8) | $(A \times B) \rightarrow C \dashv\vdash A \rightarrow (B \rightarrow C)$ | [Schönfinkel's Law] |

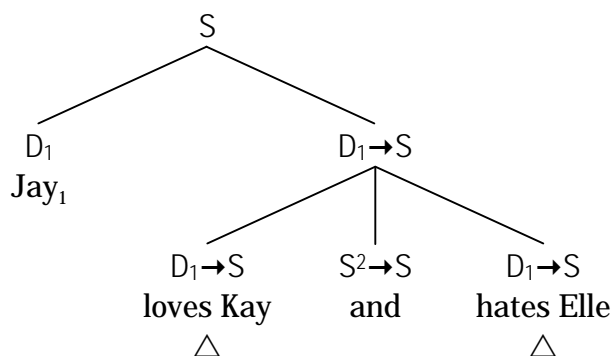
2. Examples of Non-Entailments in CGL

- | | | | |
|-----|--|----------|---------------------|
| (1) | $(A \rightarrow B) \rightarrow B \not\vdash (B \rightarrow A) \rightarrow A$ | CL-valid | [Lukasiewicz's Law] |
| (2) | $(A \rightarrow B) \rightarrow A \not\vdash A$ | CL-valid | [Peirce's Law] |
| (3) | $\emptyset \not\vdash A \rightarrow A$ | LL-valid | [Tautology] |
| (4) | $(A \rightarrow A) \rightarrow B \not\vdash B$ | LL-valid | [Law of Assertion] |
| (5) | $A \rightarrow (A \rightarrow B) \not\vdash A \rightarrow B$ | RL-valid | [Contraction] |
| (6) | $A \not\vdash A \times A$ | RL-valid | [Duplication] |
| (7) | $A \not\vdash B \rightarrow A$ | IL-valid | [Positive Paradox] |
| (8) | $A \rightarrow B \not\vdash A \rightarrow (A \rightarrow B)$ | IL-valid | [Expansion] |
| (9) | $A ; B \not\vdash A$ | IL-valid | [Simplification] |

2. Generalized-Conjunction

What is missing from System L(-Id) is a way to account for generalized-conjunction, which is illustrated in the following tree.

Jay loves Kay and hates Elle



Here, $S^2 =_{df} S \times S$. What we specifically need is the validity of the following inference.

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

This is valid in System R (Relevance Logic), but not System L (Linear Logic).

¹ Linear Logic without Identity has *no tautologies*; see item (3) in Examples of non-entailments.

3. Coordinator-Promotion

Another useful, and related, construction is coordinator-promotion, which can be type-logically rendered as follows.

$$B^2 \rightarrow B \vdash (A \rightarrow B)^2 \rightarrow (A \rightarrow B)$$

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

In this connection, we propose the term ‘**coordinator**’ for any phrase of type $\mathfrak{F}^2 \rightarrow \mathfrak{F}$ [alternatively, $\mathfrak{F} \rightarrow (\mathfrak{F} \rightarrow \mathfrak{F})$], where \mathfrak{F} is any type. Both ‘and’ and ‘plus’ are examples of coordinators. Coordinator-promotion authorizes promoting any coordinator to a parallel higher-rank coordinator, which in turn can be promoted, since the rule is recursive.

Coordinator-promotion is well-known in mathematics, corresponding to the formation of **function-spaces**. For example, since numbers can be added, number-valued functions can be just as easily added "point-wise"; in particular:

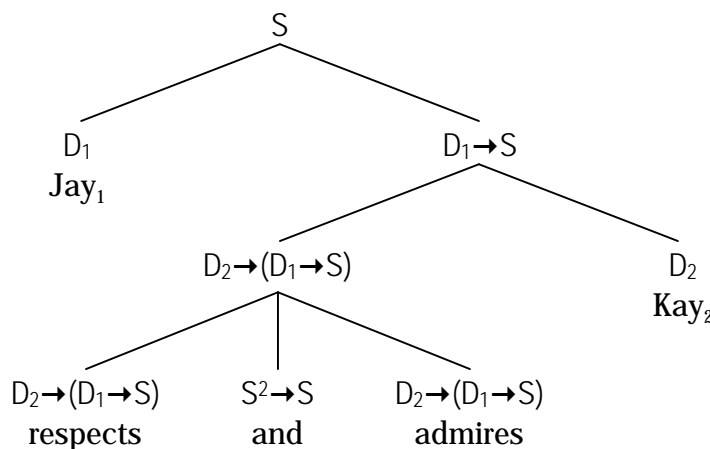
$$(f+g)(x) \quad =_{df} \quad f(x) + g(x)$$

Once again, coordinator-promotion is valid in System R, but not System L. System G (see chapter ‘Categorical Logic’) strives to achieve the proper middle ground between R and L.

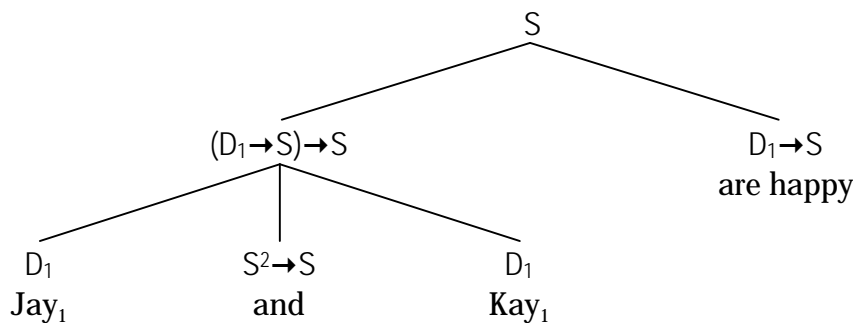
4. Example Phrase-Structures

1. Generalized Conjunction

1. Jay respects and admires Kay

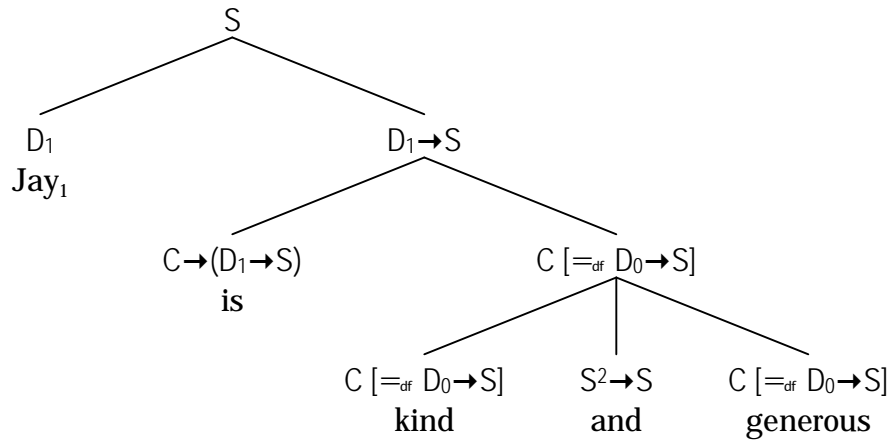


2. Jay and Kay are happy



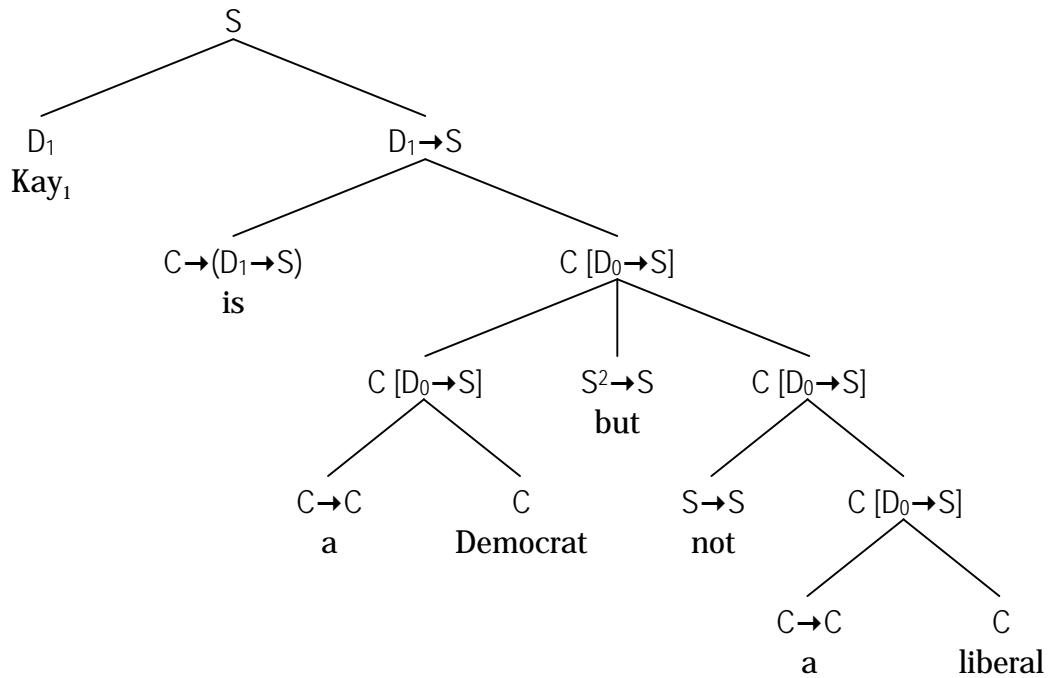
This is a fairly tricky application of CGL.

3. Jay is kind and generous



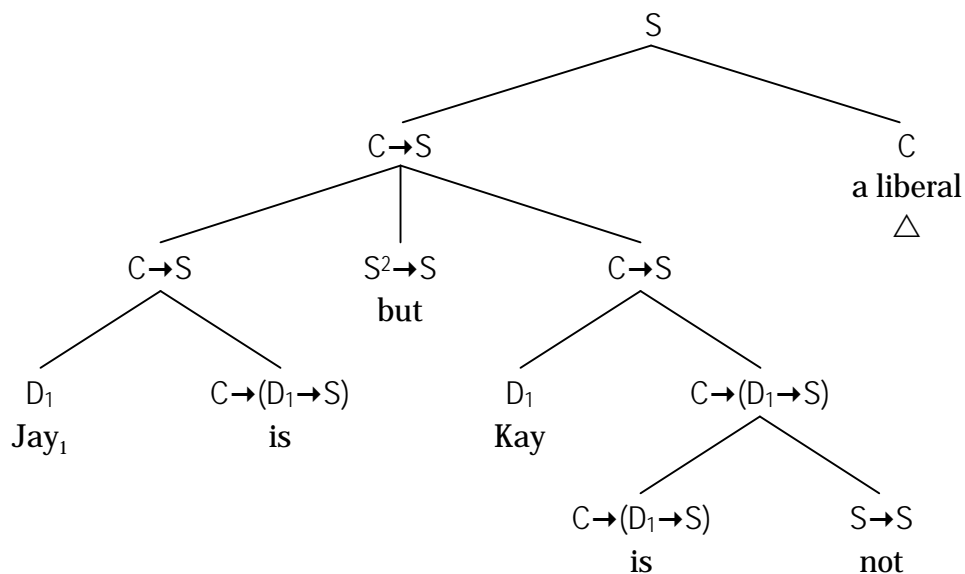
Note that the type C is henceforth defined as a 0-inflected predicate [D₀→S], which makes common-nouns and bare-adjectives a special kind of predicate. We will continue to use 'C' as shorthand for 'D₀→S'.

4. Kay is a Democrat, but not a liberal



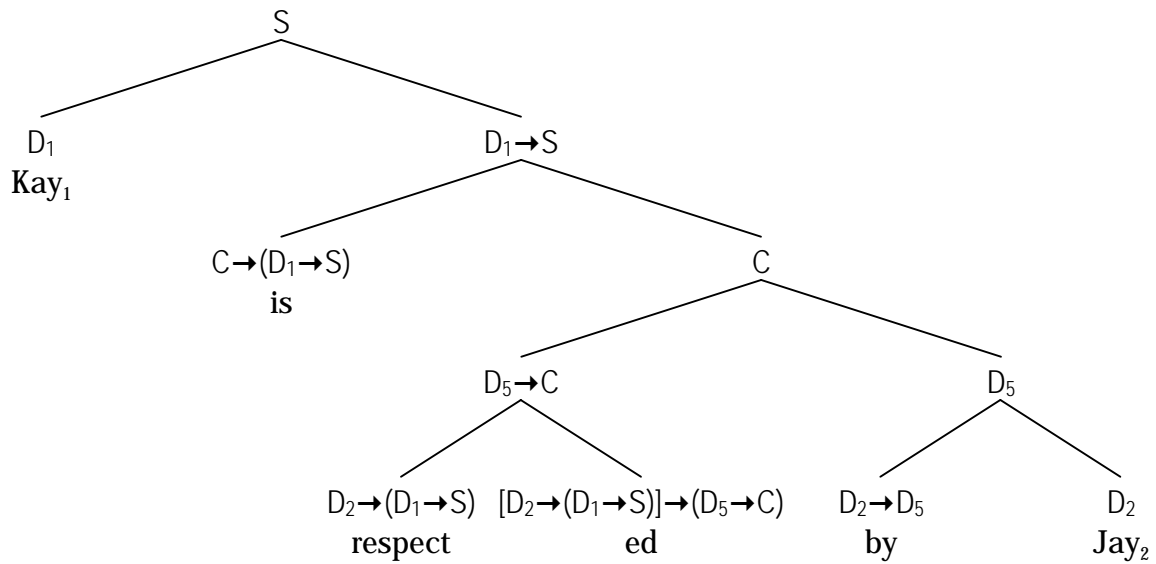
Note: we treat the indefinite article as a modifier that marks *number*, in this case *singular*. Semantically, it is equivalent to 'one' [many languages do not distinguish 'a' and 'one'].

5. Jay is, but Kay is not, a liberal

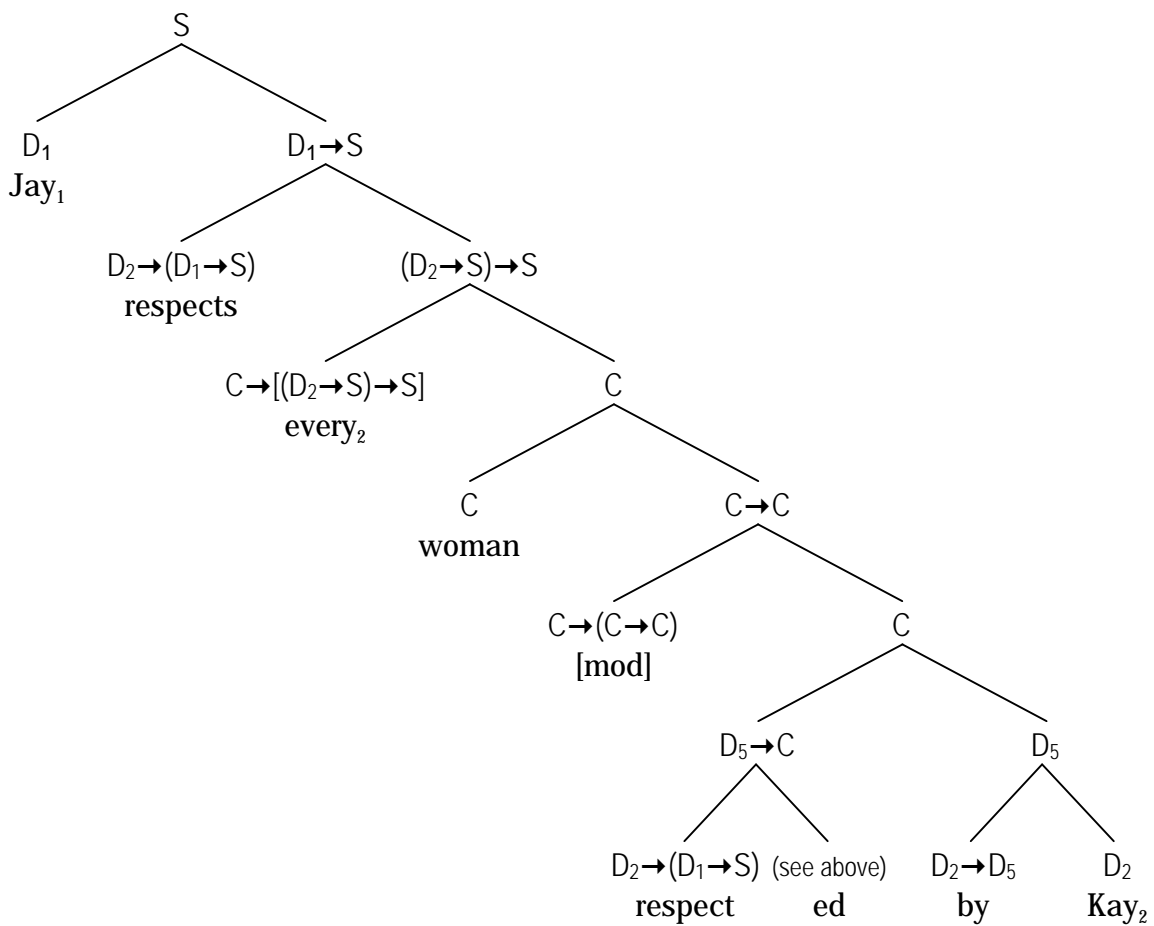


2. Passive Constructions

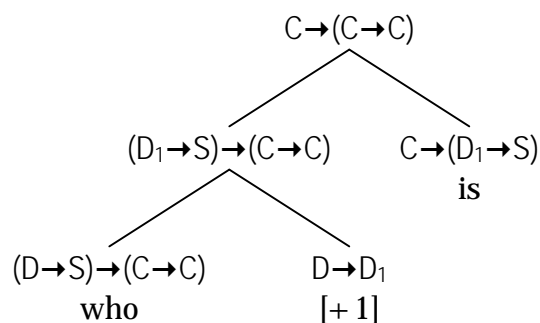
1. Kay is respected by Jay



2. Jay respects every woman respected by Kay



Note that the mod-operator is equivalent to 'who is', which is type-analyzed as follows.



Also, note that we relax the fundamental syntactic-rule that case-inflections apply only to NPs, and "pre-inflect" 'every', since eventually it is going to produce an NP. We can do the same thing with 'the' to produce 'the₁'. This is, ultimately, merely to save space in the tree.

5. Genitive-Nouns

- | | | |
|-----|-------------------------|---|
| (1) | ordinary common-nouns | denote sets of entities |
| (2) | relational common-nouns | denote relations over entities |
| (3) | genitive-nouns | relational nouns with genitive morphology |
| (4) | function-like nouns | genitive-nouns with implicit definiteness |

Genitive-nouns include

mother, father, capital, square, square-root²

brother, sister, killer, friend, citizen, member, part

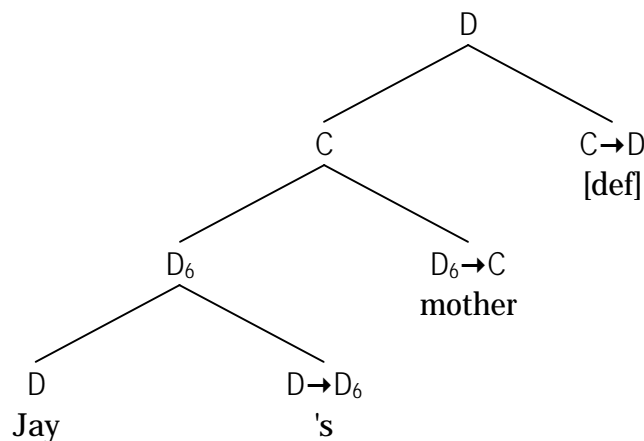
The first ones usually carry implicit definiteness, *but not always*. The latter usually do not carry implicit definiteness, but do *sometimes*. For this reason, we propose to make definiteness *morphologically-explicit*, by proposing a **definiteness-operator**, [def], which has type $C \rightarrow D$, and basically means ‘the’.

1. Initial Analysis

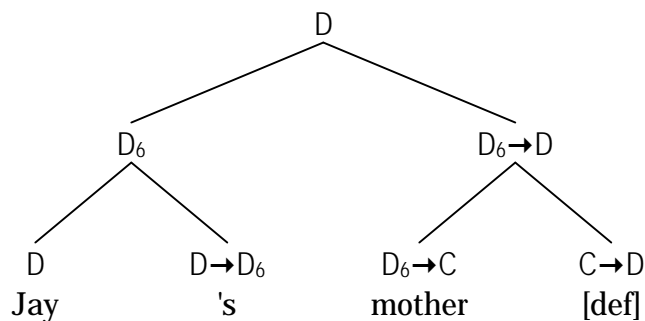
$$\begin{aligned} \text{type}(\text{GN}) &= D_6 \rightarrow C && [\delta = \text{genitive case-marker}] \\ \text{type}([\text{def}]) &= C \rightarrow D \end{aligned}$$

2. Examples

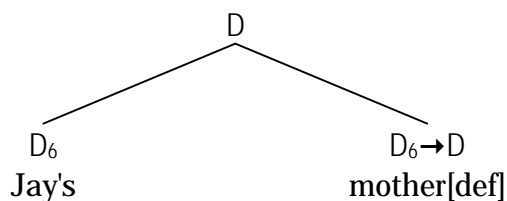
1. Jay's mother



The following is an alternative account.

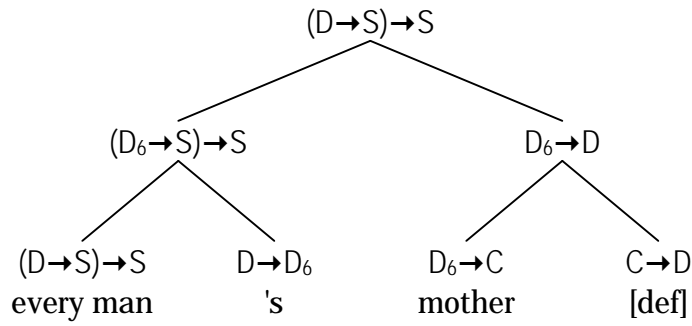
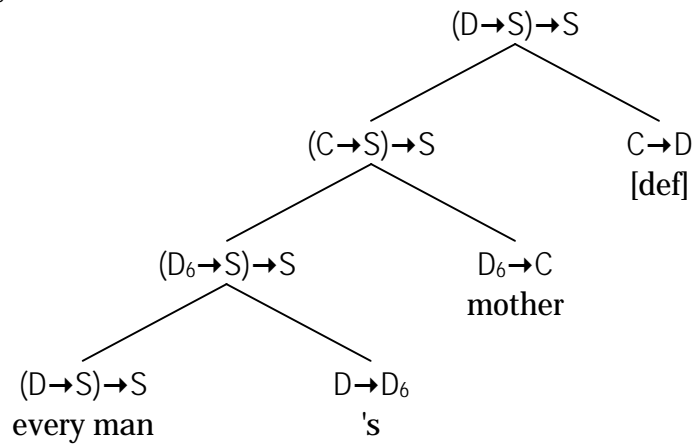


This may be streamlined as follows.

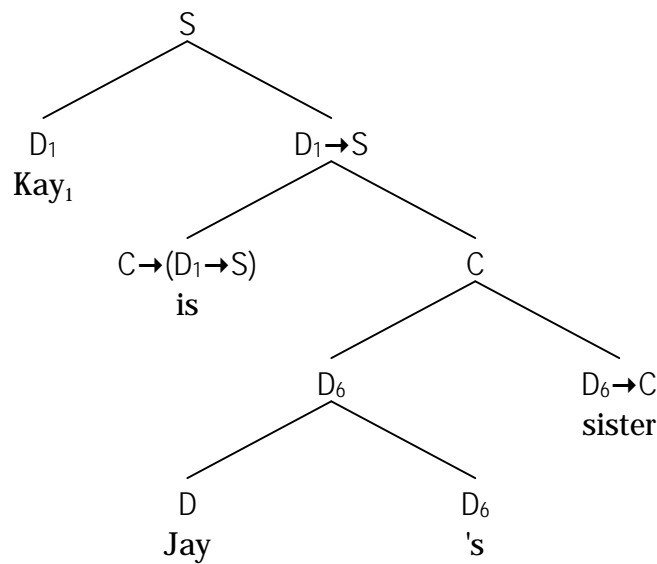


² Many genitive-nouns give rise to derivative non-relational nouns. For example, to be a mother (*simpliciter*) is to be someone's mother, and to be a killer (*simpliciter*) is to be a someone's killer.

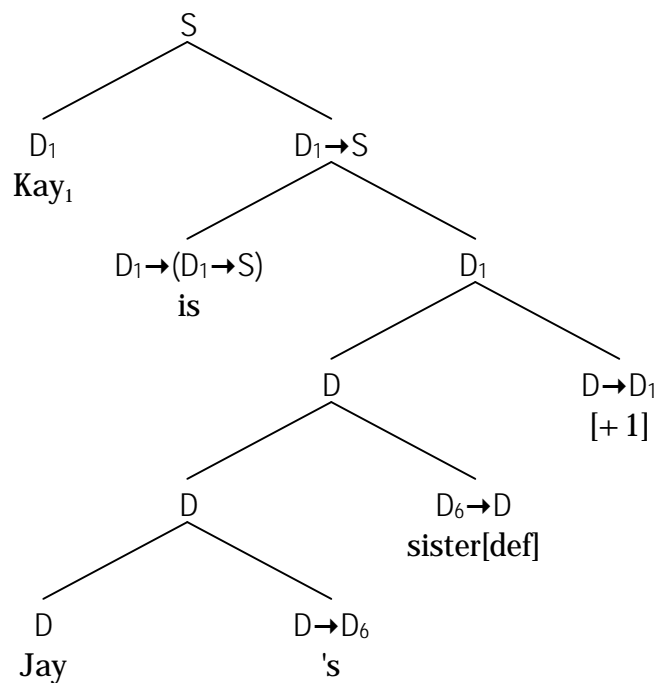
2. every man's mother



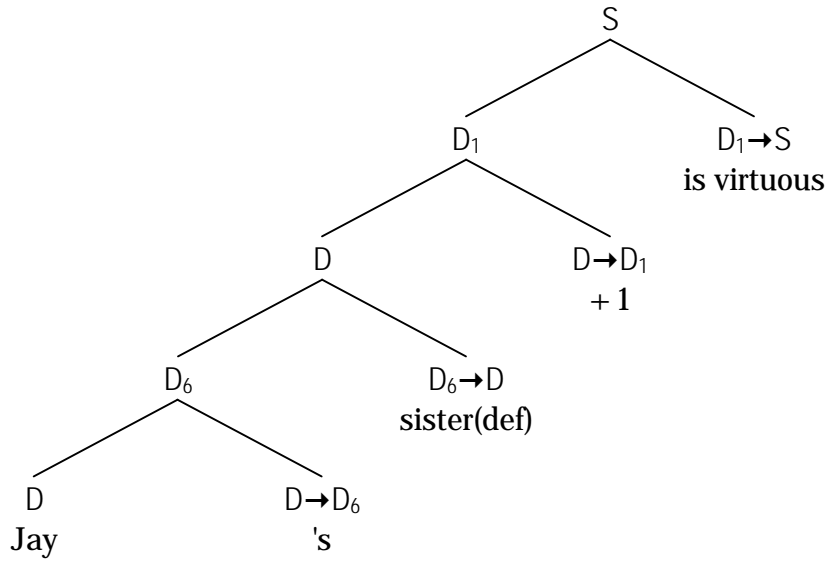
3. Kay is Jay's sister [indefinite reading]



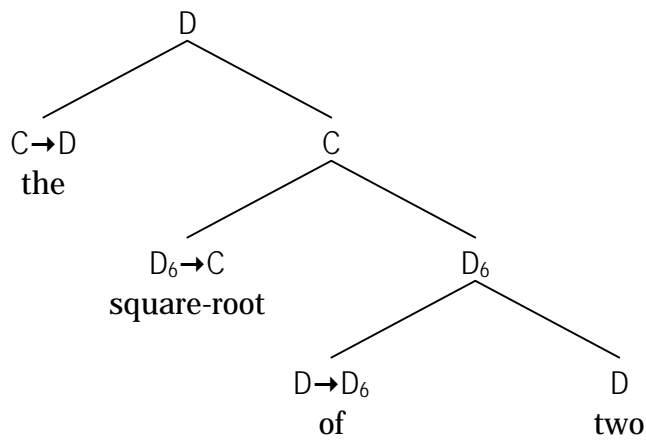
4. Kay is Jay's sister [definite reading]



5. Jay's sister is virtuous [definite reading]

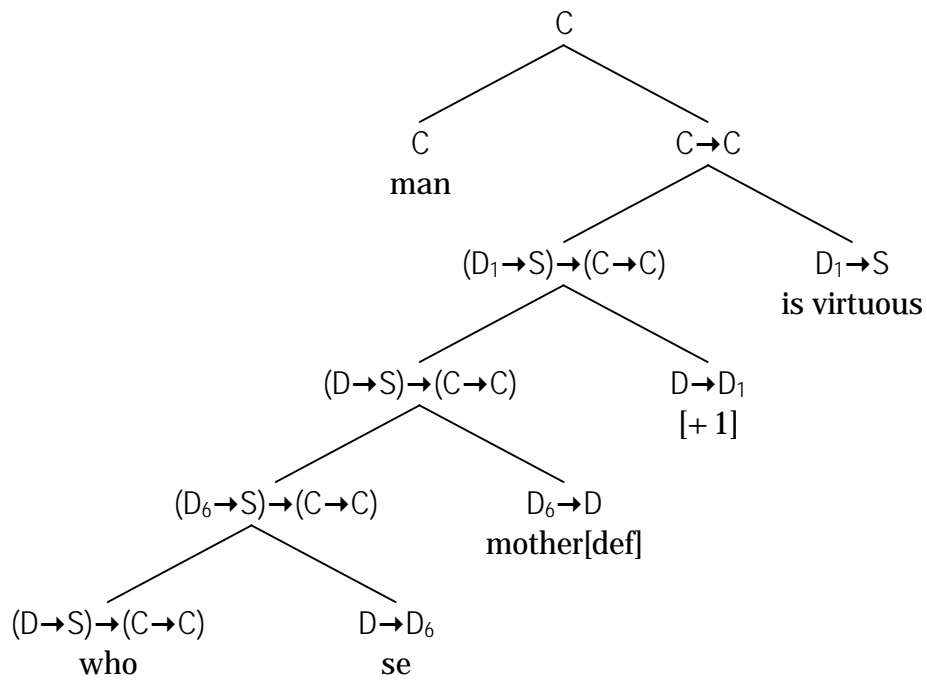


6. the square-root of two

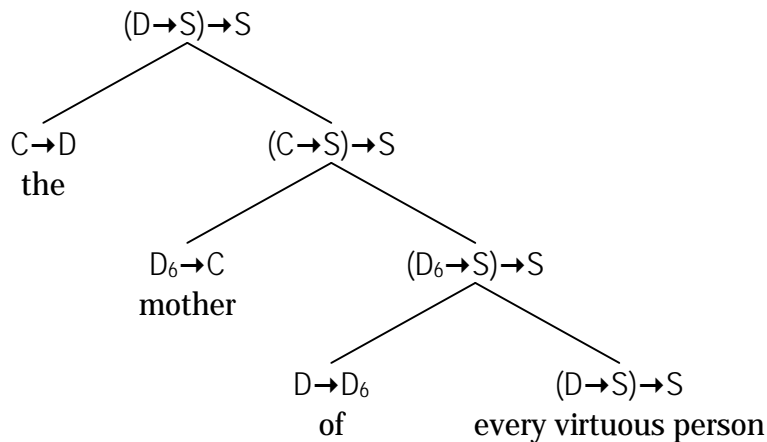
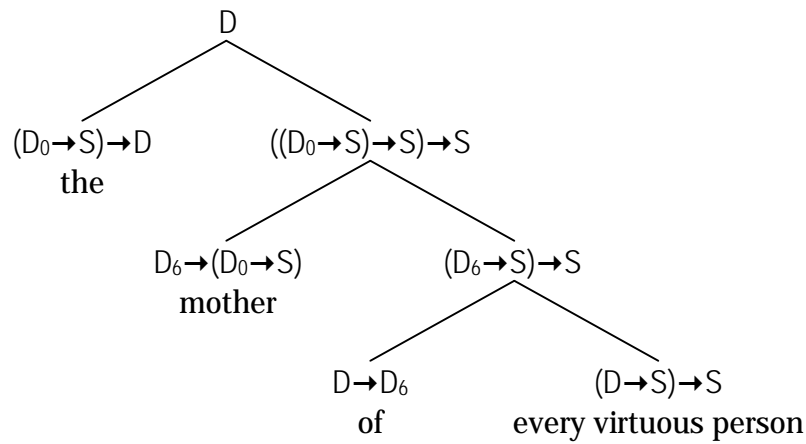


Note: here, definiteness is morphologically overt, being 'the'.

7. man whose mother is virtuous



8. the mother of every virtuous person [very tricky!]



6. Possessives

The possessive-construction is syntactically just like the genitive-construction in many languages; for example, in English, they both have two forms – ‘of’ and apostrophe-‘s’. But semantically, possessive and genitive are quite different. In both cases, a relation is involved; however, in the case of genitive; the relation is *internal* to the noun, whereas in the case of possessive, the relation is *external* to the noun. For example, the relation between Jay and his mother is contained in ‘mother’, whereas the relation between Jay and his dog is not contained in ‘dog’.

The following is a categorial account of possessive.

$$\text{type(pos)} = D \rightarrow (C \rightarrow C)$$

Also, just like genitive, possessive can be definite or indefinite. The following is an example analysis, according to which the sentence is non-committal about how many dogs Kay owns.

Rufus is Kay's dog [indefinite reading]

