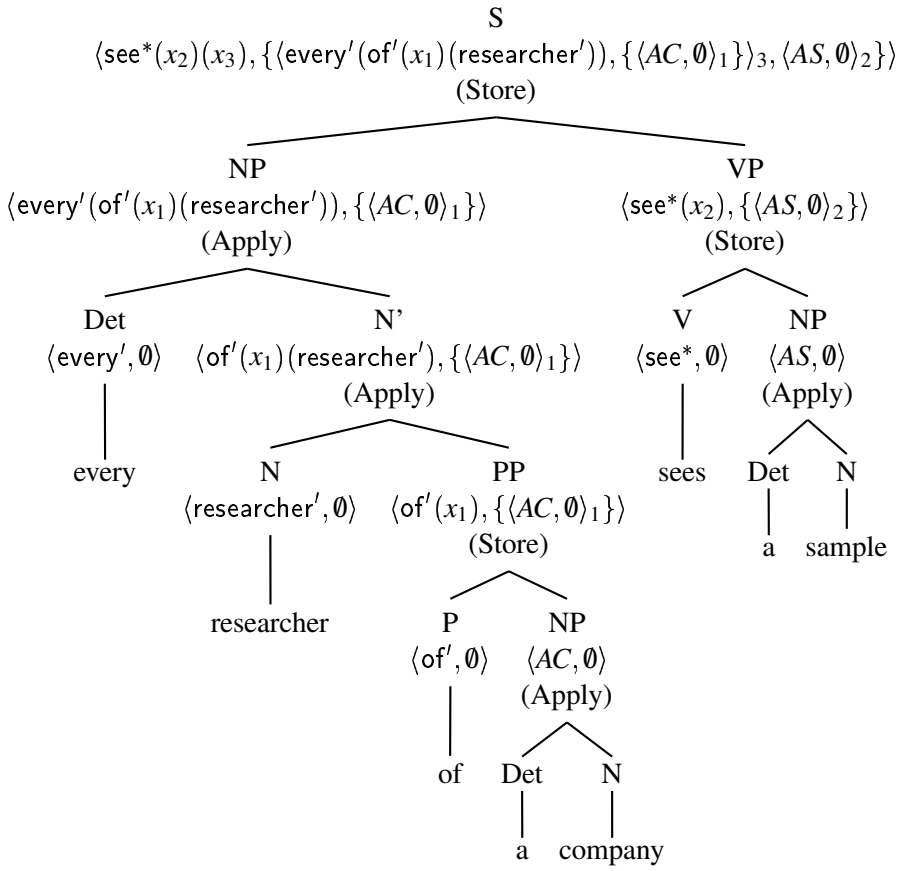
4 Nested Cooper Storage

The syntax tree, decorated with the semantic values for each syntax node, is shown below. For each node, we have indicated which rule (Store, Retrieve, or Apply) was used to derive the semantic value from the semantic values of the children.

We use the following abbreviations:

abbreviation	full term	type
AC	$\lambda P \exists x.(\mathsf{company}'(x) \wedge P(x))$	$\langle\langle e,t\rangle,t\rangle$
AS	$\lambda P \exists x.(\mathsf{sample}'(x) \wedge P(x))$	$\langle\langle e,t\rangle,t\rangle$
ER	$\lambda P \forall x.(\mathsf{researcher}'(x) \wedge \mathsf{of}^*(x_1)(x)) \rightarrow P(x)$	$\langle\langle e,t\rangle,t\rangle$
of'	$\lambda x \lambda F \lambda y.\mathsf{of}^*(y)(x) \wedge F(y)$	$\langle e,\langle\langle e,t\rangle,\langle e,t\rangle\rangle\rangle$
every'	$\lambda P \lambda Q \forall x.P(x) \rightarrow Q(x)$	$\langle\langle e,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle$



Note that we almost never have two different rules which could be applied at the same node. The only exception is the root sentence node, at which we could apply either Store or Functional Application. We have shown the result of the Store rule, but the Functional Application rule would eventually yield exactly the same three readings. In particular, be aware the Storage rule is only applicable if one of the syntactic children is an NP and the other child is of type $\langle e, \tau \rangle$ for some τ, so both the N' and the subject NP node must get exactly the given semantic value.

The complex term $\mathsf{of}'(x_1)(\mathsf{researcher}')$, which occurs in the store entry 3, beta-reduces to the following term:

$$\begin{array}{ll} & \mathsf{of}'(x_1)(\mathsf{researcher}') \\ \rightarrow^*_\beta & \lambda P \forall x.(\mathsf{researcher}'(x) \wedge \mathsf{of}^*(x_1)(x)) \rightarrow P(x) \end{array}$$

We abbreviate this term as AR (see table above).

Now we can turn to the retrieval steps. We can generate three readings of the sentence by retrieving quantifiers from the store in different orders.

Reading 1:

$$\begin{array}{ll} & \langle \mathsf{see}^*(x_2)(x_3), \{\langle ER, \{\langle AC, \emptyset \rangle_1\}\rangle_3, \langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R2} & \langle AS(\lambda x_2.\mathsf{see}^*(x_2)(x_3)), \{\langle ER, \{\langle AC, \emptyset \rangle_1\}\rangle_3\}\rangle \\ \Rightarrow_{R3} & \langle ER(\lambda x_3.AS(\lambda x_2.\mathsf{see}^*(x_2)(x_3))), \{\langle AC, \emptyset \rangle_1\}\rangle \\ \Rightarrow_{R1} & \langle AC(\lambda x_1.ER(\lambda x_3.AS(\lambda x_2.\mathsf{see}^*(x_2)(x_3)))), \emptyset\rangle \\ \rightarrow^*_\beta & \exists x.\mathsf{company}'(x) \wedge \forall y.(\mathsf{researcher}'(y) \wedge \mathsf{of}^*(x)(y)) \rightarrow \exists z.(\mathsf{sample}'(z) \wedge \mathsf{see}^*(z)(y)) \end{array}$$

Reading 2:

$$\begin{array}{ll} & \langle \mathsf{see}^*(x_2)(x_3), \{\langle ER, \{\langle AC, \emptyset \rangle_1\}\rangle_3, \langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R3} & \langle ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3)), \{\langle AC, \emptyset \rangle_1, \langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R1} & \langle AC(\lambda x_1.ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3))), \{\langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R2} & \langle AS(\lambda x_2.AC(\lambda x_1.ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3)))), \emptyset\rangle \\ \rightarrow^*_\beta & \exists z.\mathsf{sample}'(z) \wedge (\exists x.\mathsf{company}'(x) \wedge \forall y.(\mathsf{researcher}'(y) \wedge \mathsf{of}^*(x)(y)) \rightarrow \mathsf{see}^*(z)(y)) \end{array}$$

Reading 3:

$$\begin{array}{ll} & \langle \mathsf{see}^*(x_2)(x_3), \{\langle ER, \{\langle AC, \emptyset \rangle_1\}\rangle_3, \langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R3} & \langle ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3)), \{\langle AC, \emptyset \rangle_1, \langle AS, \emptyset \rangle_2\}\rangle \\ \Rightarrow_{R2} & \langle AS(\lambda x_2.ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3))), \{\langle AC, \emptyset \rangle_1\}\rangle \\ \Rightarrow_{R1} & \langle AC(\lambda x_1.AS(\lambda x_2.ER(\lambda x_3.\mathsf{see}^*(x_2)(x_3)))), \emptyset\rangle \\ \rightarrow^*_\beta & \exists x.\mathsf{company}'(x) \wedge (\exists z.\mathsf{sample}'(z) \wedge \forall y.(\mathsf{researcher}'(y) \wedge \mathsf{of}^*(x)(y)) \rightarrow \mathsf{see}^*(z)(y)) \end{array}$$

Notice that the readings 2 and 3 are logically equivalent; they only differ in the relative scope of the two existential quantifiers. The ambiguity between these two different but equivalent readings is called *spurious*.

Notice also that in all three readings, "a company" takes scope over "every researcher", i.e. all three readings say that there is one specific company, and we are only talking about researchers who are working for this particular company. This is because the nesting structure of the quantifier store forces us to retrieve and apply the entry 3 before we can access the entry 1. The (quite plausible) reading in which, say, each researcher who works for *any* company saw some (different) sample (i.e. "every researcher" outscopes both "a company" and "a sample") can't be generated by the version of Nested Cooper Storage we presented in the course.

The problem is that because we analysed of' as a term of type $\langle e, \langle \langle e,t\rangle, \langle e,t\rangle\rangle\rangle$, we *must* combine it with the NP "a company" by applying the Store rule. We can get the two missing readings if we assume the following semantic representation for the word "of":

$$\mathsf{of}'_2 = \lambda Q \lambda F \lambda y.F(y) \wedge Q(\mathsf{of}^*(y))$$

This term is of type $\langle \langle \langle e,t\rangle, t\rangle, \langle \langle e,t\rangle, \langle e,t\rangle\rangle\rangle$, i.e. it is a type-raised version of of' above. Now we can apply Functional Application to combine the semantic representations of "of" and "a company", and this gives us the two other readings. (Please check this yourself!) But because we now actually *must* combine these constituents by Functional Application and cannot apply the Store rule, we miss the three readings that we got with the original analysis of "of".

The original version of Nested Cooper Storage, as presented by Keller (1988) doesn't have this problem. It only occurs because we have (over-)simplified the presentation of NCS in the lecture.