Basic syntax of quantification in English

Jeff Speaks
PHIL 43916

September 15, 2014

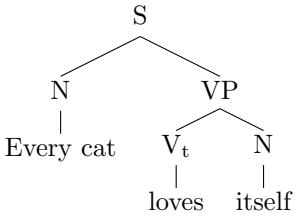
1 The problem posed by quantified sentences

So far the language for which we have developed a semantics contains no devices for expressing generality, which we accomplish in English by expressions like 'someone,' 'everything', or 'most dogs.'

At first glance, it is not easy to see how to think about the structure of such sentences. Consider, for example,

> Every cat loves itself.

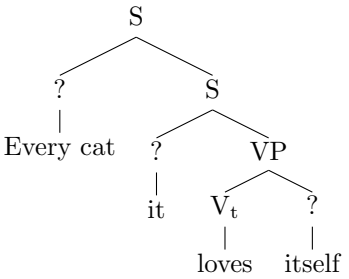
It looks like 'Every cat' is playing the same role as a standard N, and 'loves' looks like a V_t. Hence one might think that the right tree diagram for this sentence is something like



But it's hard to see how this is going to work, for a few reasons. First, it is not obvious what could be given as $[\![\text{Every cat}]\!]$ – our previous semantics leads us to expect that this will be some individual, but which one could it be? (You might say: the set of all the cats. But this would make a mess of our previous semantic theories, since then the sentence would be true only if a certain set loves itself – and sets are not the sorts of things which love anything.)

Second, even if we could solve this problem, we have no way to account for the fact that the interpretation of 'itself' seems somehow linked to 'every cat'; our previous theories give us no way to represent the idea that the semantic value of one node might depend in some way on the semantic value of another, unless one is a parent of the other.

Problems involving the logic and semantics of quantified sentences were a source of intensive study from ancient times through the Middle Ages. The principal breakthrough in their treatment was due to Frege. One way to think of Frege's basic idea is to think of him as saying that 'Every cat loves itself' does not have the form described by the tree above, but rather a form something like



which introduces quantifier expressions like 'every cat' and variable expressions like 'it' as new syntactic categories. The idea is that we understand the function of expressions like 'every cat' not as simple N's but rather as 'saying something about' a sentence, like 'it loves itself', which contains variable expressions. Intuitively, what it says about such a sentence is that, if we restrict ourselves to the cats, then every assignment of an object as the semantic value of 'it'/'itself' yields a true sentence.

Our job now is to make this idea precise, and to see how to apply it to an expanded fragment of English. We begin by asking how to understand the syntactic form of sentences containing quantifier expressions.

2 Basic syntactic categories for quantification in English

In the logic, our expressions for expressing quantification are '$\forall$' and '$\exists$.' But in English we express ourselves using complex phrases like 'every cat' or 'a dog.' (Even 'everything' can be viewed as a composite of 'every' and 'thing.') In quantifier phrases like this, we separate out two parts: the determiner ('every') and the restrictor ('cat'). These are new syntactic categories:

Det (determiner) $\rightarrow$ a, every, the

N_c (common noun) $\rightarrow$ book, fish, man, woman

You get a quantifier phrase by combining a Det with a N_c.

These quantifier phrases are themselves of the syntactic category NP (noun phrase). (We're leaving the category N, from our earlier language, out of this one; we'll reintroduce it later.)

This tells us the categories to which quantifier phrases and their parts belong; but it does not yet tell us how these quantifier phrases combine with other expressions to form sentences.

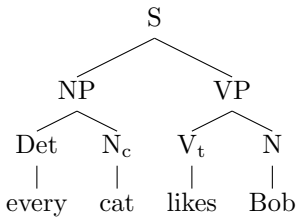
3 Quantifier movement

In the sentences we have discussed so far, the logical form of the sentence – its LF – mirrors its surface structure. So, for example, the order of the leaves of the tree from left to right has been the same as the order of the words in the sentence from left to right. Quantifier phrases force us to complicate things, and posit a greater difference between surface structure and LF than we have encountered so far.

To see why this is needed, let's suppose that we tried to construct trees for quantified sentences while letting those trees mirror surface form. Then the tree for the sentence

Every cat likes Bob.

would presumably be something like

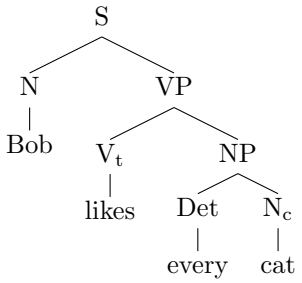


Now think informally about what the meaning of 'every' could be. It seems to be, roughly, a function from a pair of sets to a truth-value. In this case, the sets are the set of cats and the set of things that like Bob, and the sentence will be true if every element of the first set is also a member of the second. (In the above tree, we would model this by 'currying' this function, and letting $[\![\text{every}]\!]$ be a function from sets to a function from sets to truth-values.)

Presumably 'every' should mean the same thing in the case of our first sentence,

Bob likes every cat.

It's tree, if we are letting LF mirror surface structure, will presumably be something like



But what should the two sets be in that case? Presumably, the set of cats, and the set of things Bob likes. But look at the above tree. What node will have as its semantic value the set of things Bob likes?

(Note: you could get around this problem by positing a lexical ambiguity in 'every', and try to use this to interpret quantifiers 'in situ' rather than via quantifier movement. The extra reading for today gives some arguments in favor of quantifier movement.)

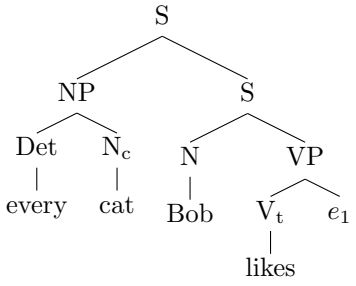
To give a unified treatment of 'every cat' in these two sentences, we posit *quantifier movement* or *quantifier raising*. The idea, intuitively, is that the quantifier 'moves' from its place in the surface structure to a higher level of our tree for that sentence. When a constituent is moved, it leaves behind a *trace* in the LF of the sentence.

Traces are a bit like variables. Our language will contain infinitely many traces e_1, e_2, ... The number which follows a trace is its 'index.'

The rule for quantifier raising is a transformation – a rule which tells you what trees might be associated with what surface structures. It is stated as:

(57) $[_S$ X NP Y$] \Rightarrow [_S$ NP$_i$ $[_S$ X e_i Y$]$]

Given this, our tree for 'Bob likes every cat' is not the one given above, but rather



This also tells us how to form sentences out of quantifier phrases. These NP's are expressions which, like sentence operators, combine with sentences to form sentences.

It is worth noting that while quantified sentences are one case which force us to posit divergence between surface structure and LF, they are not the only such case. Another is one we have already encountered:

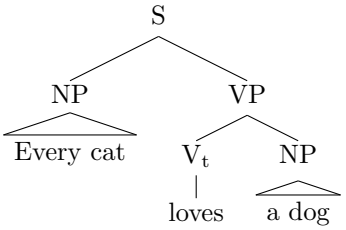
Bob is easy to please.

Bob is eager to please.

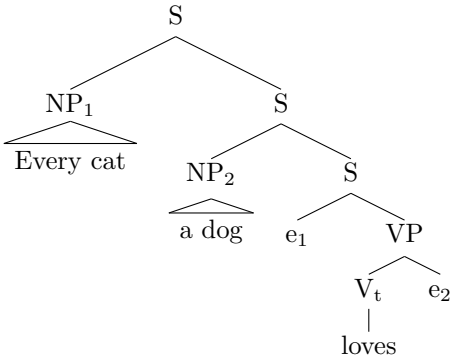
Despite the similarity of their surface forms, these sentences seem to have very different structures. One might explain this by taking them to be transformations of very different logical forms. We'll return to these sentences later in the course.

4 Binding and multiply quantified sentences

Some sentences, like 'Every cat loves a dog' contain multiple devices of generality. Remember that rather than thinking of such sentences as of the form



we are thinking of them instead as of the form



But we need some way of representing the fact that 'Every cat' is connected to 'e_1' whereas 'a dog' is connected to 'e_2'. (We'll spell out what this connection amounts to semantically in a bit.) We express this connection by saying that 'Every cat' *binds* 'e_1' whereas 'a dog'*binds* 'e_2'. A trace which is bound by some quantifier is a 'bound trace.' A variable which is not bound by any quantifier is a 'free trace.'

How do we tell from a tree which NP's are supposed to bind which traces? In two ways: first, by the indices on traces we use; and, second, by relative locations of traces and quantifiers in the relevant tree structure. A quantifier binds a trace iff (i) they are coindexed and (ii) the quantifier *C-commands* the trace. The crucial notion of C-command is defined as follows:

> A C-commands B iff the first branching node that dominates A also dominates B.