Quantification in the predicate calculus

PHIL 43916
September 5, 2012

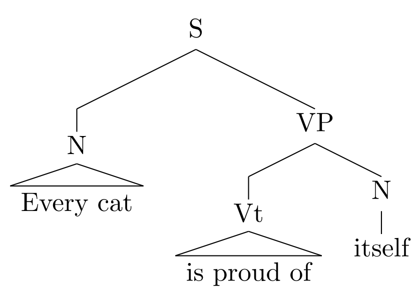
1. The problem posed by quantified sentences

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 is proud of itself.

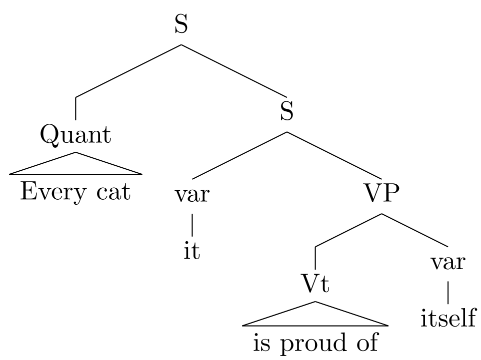
It looks like 'Every cat' is playing the same role as a standard N, and 'is proud of' looks like a V_t. Hence one might think that the right tree diagram for this sentence is something like



But it's hard to see how thius is going to work, for a few reasons. First, it is not obvious what could be given as ⟦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 was proud of itself — and sets are not the sorts of things which can be proud.)

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 expression might depend in some way on the semantic value of another, unless one dominates 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 is proud of itself’ does not have the form described by the tree above, but rather a form something like



which introduces quantifiers and variables 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 is proud of 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 to ‘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. But it’s useful, as in the text, to begin by looking at how we treat quantification in the predicate calculus.

2. Syntax of PC

My presentation of this will differ a bit from the way it is done in the text; I will stick with the (now) familiar categories of S, N, and conj rather than replacing them with Form (for formula), const (for constant), and Conn (for connective). I will, however, simplify by following the text in replacing VPs with Preds (predicates).

We add the category t (for term). There are two types of terms: names (N) and variables (var).

We admit the subcategories Pred_1, Pred_2, etc. where the subscript represents the number of terms that the predicate in question must combine with in order for form a S. We have:

> $\text{Pred}_1 \rightarrow$ is boring, is hungry
> $\text{Pred}_2 \rightarrow$ likes, $=$
> $\text{Pred}_3 \rightarrow$ _ introduces _ to _.

In addition we add to our lexicon the following expressions:

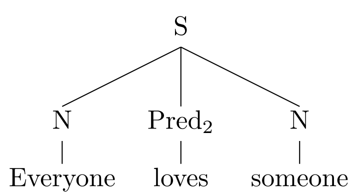
> $\rightarrow$ ('if ... then')
> $\leftrightarrow$ ('if and only if')
> $\forall$x ('every')
> $\exists$x ('some')
> x_1, x_2, ... (the variables)

The sentences of our language are of the forms

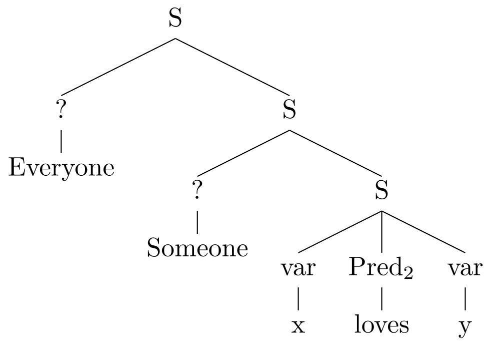
> S conj S
> neg S
> Pred_n (t_1 ... t_n)
> $\forall$x S
> $\exists$x S

3. Bound and free variables

Some sentences, like 'Everyone loves someone' 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 (something like) the form

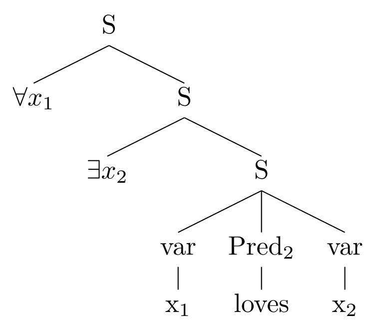


But we need some way of representing the fact that 'Everyone' is connected to 'x' whereas 'Someone' is connected to 'y'. (We'll spell out what this connection amounts to in a bit.) We express this connection by saying that 'Everyone' *binds* 'x' whereas 'Someone' binds 'y'. A variable which is bound by some quantifier is a 'bound variable.' A variable which is not bound by any quantifier is a 'free variable.'

In the syntax of PC as described above, how do we indicate what binds what? In two ways: first, by the indices on variables we use; and, second, by relative locations of variables and quantifiers in the relevant tree structure. We say that a quantifier binds a variable iff (i) they are coindexed and (ii) the quantifier c-commands the variable, where

> A c-commands B iff neither A nor B dominate each other, and the first branching node that dominates A also dominates B.

We can then make clear what binds what in the above tree by changing it to



4. Models and assignments

What we need to know now is how to interpret trees like this: what does it take for them to be true or false?

Two answer this question we need to introduce two new concepts: the concept of a *model* and the concept of an *assignment*.

A model M is a pair of two things: a valuation function V and a domain (or universe) U of discourse. These ideas are already familiar, in somewhat different form, from our language F_1. Recall that we needed to talk not just about, for example, ⟦is boring⟧, but also ⟦is boring⟧^{v} — where the latter is ⟦boring⟧ relative to some circumstance of evaluation. A valuation function V_1 is a function from the expressions of our language to their semantic value in a situation v_1.

A domain U_1 is just the list of things that exist in v_1. The semantic values assigned to expressions by V_1 must be built up from elements of U_1. In the examples we've been discussing, the domain was {Pavarotti, James Bond, Sophia Loren}.

So we can talk about the semantic value of "is boring" relative to a model in much the way we before talked about ⟦is boring⟧^{v}.

An assignment g is a function from variables to elements of the domain. Recall that we have infinitely many variables x_1, x_2, ... One assignment, given the above domain of discourse, might be

[$x_1 \rightarrow$ James Bond
$x_2 \rightarrow$ Sophia Loren
$x_n \rightarrow$ Pavarotti] for any $n>2$.

Another might be

[$x_1 \rightarrow$ James Bond
x_2, $x_3 \rightarrow$ Pavarotti
$x_n \rightarrow$ Sophia Loren] for any $n>3$.

A very simple one might be

[$x_n \rightarrow$ James Bond] for any n

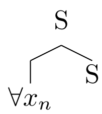
All that's required is that the assignment g be a function from the variables to members of the domain. We can now talk about the semantic value of an expression relative to a model and an assignment (written as, e.g., ⟦is boring⟧$^{M1,\, g1}$). This notion is defined (for the lexicon described above) as follows:

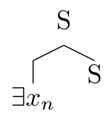
For a model $M_1 = <U_1, V_1>$,
if A is an expression which is not a variable — e.g., a N or a Pred — then
$[\![A]\!]^{M1,\, g1} = V_1(A)$; and

if A is a variable, then $[\![A]\!]^{M1,\, g1} = g_1(A)$.

5. Interpretation of sentences containing one quantifier

Our language contains two sorts of sentences containing quantifiers:





What does it take for these sentences to be true or false?

Consider the first, universally quantified, sentence first. The basic idea is that we start with a model *M* and an assignment *g* of values to all of the variables, and we then consider every assignment function which agrees with *g* on every variable *other than* x_n. If S is true in *M* with respect to *every* such assignment, then our quantified sentence is true relative to *M* and *g*.

We use

$g_1\ [u/x_n]$

to mean

the assignment function which differs from g_1 only in assigning u as the value of x_2

We can then write out our truth conditions as:

(21a) $[\![\forall x_n\ A]\!]^{M1,\, g1} = 1$ iff for all $u \in U_1$, $[\![A]\!]^{M1,\, g1\ [u/xn]} = 1$

Existential quantification is treated in a parallel way:

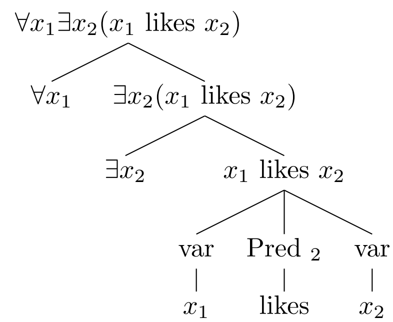
(21b) $[\![\exists x_n\ A]\!]^{M1,\, g1} = 1$ iff for some $u \in U_1$, $[\![A]\!]^{M1,\, g1\ [u/xn]} = 1$

Given our domain of three individuals and the choice of the first assignment function described above, how would you derive the truth value for '∃x x is hungry'?

Did it matter which assignment function we picked?

6. Multiply quantified sentences and scope

Let's see how this helps us to derive truth conditions for sentences with more than one quantifier. On one reading, the sentence 'Everyone likes someone' might be written out in our present language as '$\forall x_1\ \exists x_2\ (x_1$ likes $x_2)$', with the tree diagram



We first apply rule (21a), and get the result that the sentence is true iff for every $u \in U$, $[\![\exists x_2\ (x_1$ likes $x_2)]\!] = 1$ relative to M_1 and $g_1[u/x_1]$ — i.e., if, for every member of the domain, the sentence is true relative to the assignment which differs from g_1 only in assigning u as the value of x_1.

How do we tell whether it is? What we want to do is apply rule (21b) to $\exists x_2\ (x_1$ likes $x_2)$. But this contains the variable 'x_1' which, in *this* sentence, appears to be free. What do we do with it?

The answer is that we were already told what to do in the previous step. We know that for the sentence as a whole to be true, we need '$\exists x_2\ (x_1$ likes $x_2)$' to be true relative to, for all $u \in U$, $g_1[u/x_1]$. That means that we have to consider (given our small domain) three cases: one in which Pavarotti $= [\![x_1]\!]$, one in which Bond $= [\![x_1]\!]$, and one in which Loren $= [\![x_1]\!]$. The whole sentence is true iff the sub-sentence '$\exists x_2\ (x_1$ likes $x_2)$' is true relative to each of these assignments.

How do we tell whether it is? Now we can apply rule (21b) — three times, corresponding to the three different values of $[\![x_1]\!]$. Let's focus first on the case in which Pavarotti $= [\![x_1]\!]$. What rule (21b) tells us is that '$\exists x_2\ (x_1$ likes $x_2)$' is true relative to this assignment iff for some $u' \in U$, 'x_1 likes x_2' is true relative to g_1[Pavarotti$/x_1$[u'$/x_2]\!]$ — i.e., iff, for some member u' of the domain, it is true relative to the assignment of values to variables which differs from g_1 only in assigning Pavarotti as the value of x_1 and u' as the value of x_2.

We then go through the same procedure for assignments in which Bond $= [\![x_1]\!]$, and in which Loren $= [\![x_1]\!]$. If '$\exists x_2\ (x_1$ likes $x_2)$' comes out true for all, then our original sentence '$\forall x_1\ \exists x_2\ (x_1$ likes $x_2)$' is true (relative to the model and assignment).

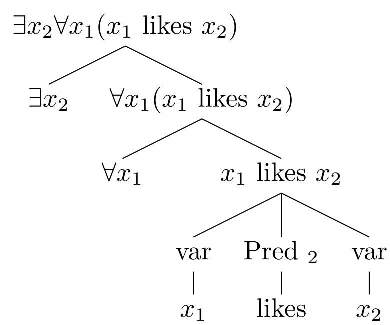
The process is not quite this tiresome if we want to derive truth conditions rather than truth-value. Then rules (21a-b) let us derive the result that

$[\![\forall x_1 \, \exists x_2 \, (x_1 \text{ likes } x_2)]\!]^{M,\, g} = 1$ iff for all u ∈ U and some u' ∈ U,
$[\![x_1 \text{ likes } x_2]\!]^{M,\, g \; [u/x1[u'/x2]]} = 1$.

We could then of course analyze this further by looking at what our semantic theory says about sentences of the form $[_S \text{N Pred}_2 \text{ N}]$, like '$x_1$ likes x_2.'

In the tree above, the universal quantifier contains the existential quantifier within its scope. In tree diagram terms, this means that the first branching node which dominates the universal quantifier is not the first branching node which dominates the existential quantifier, but does dominate it. When this is the case we say that the universal quantifier has wide scope relative to the existential quantifier, and, conversely, that the existential quantifier has narrow scope relative to the universal quantifier.

Scope relations can have an important effect on truth conditions. Consider, for example, the tree with the scope relations of our example sentence reversed:



What would its truth conditions be?