Some basic concepts of syntax

PHIL 43916
August 27, 2012

1. Syntactic categories 1
2. Syntactic structure 2

1. Syntactic categories

Our aim is to give a compositional semantic theory for a language with infinitely many sentences: a theory which (i) assigns a meaning to each of the finite simple expressions of the language and (ii) assigns rules which tell us how the meanings of complex expressions are determined by the meanings of the simple expressions of which they are composed, plus their mode of combination.

To do this, we're going to need some rules which tell us what the complex expressions of the language are: how we're allowed to combine simple expressions into complex expressions, complex expressions into yet more complex expressions, and so on.

And to do this, we're going to need some way of categorizing expressions of the language, so that we can have rules like 'a name plus an intransitive verb is a sentence, but two names in a row isn't.'

As mentioned last time, our strategy for constructing a theory will be to begin with a very simple language, and then keep adding to it, until we get as close as we can to a natural language like English. Our initial language will have comparatively few syntactic categories.

The three simple syntactic categories will be:

N (name)
V_i (intransitive verb)
V_t (transitive verb)

In addition, we will have one *sentence operator*,

neg ("it is not the case that")

and two *sentence connectives*

and
or

A sentence operator is an expression which combines with a sentence to form another sentence, whereas a sentence connective combines with two sentences to form a sentence.

We'll have two complex syntactic categories: verb phrases (VP) and sentences (S).

A verb phrase is, intuitively, the sort of thing you can combine with a name to get a sentence. In our language, there are two ways to form a verb phrase. Can you see what they would be?

This rule is written as:

VP → V_i
VP → V_tN

There are three ways to form a sentence. What are they? How would you write the rules?

Following the example language we'll be working with from the text, we'll limit ourselves to the following vocabulary (in addition to 'and', 'or', and 'it is not the case that'):

names: Pavarotti, Sophia Loren, James Bond
V_i: is boring, is hungry, is cute
V_t: likes

Given this vocabulary, and the rules for forming sentences, how many total sentences are there in our language?

That is all you will need to know for now about these syntactic categories. Note that we have so far said nothing about what any expression *means*; we've only said things about what can be grammatically combined with what.

2. Syntactic structure

But even if this is all we need to know about syntactic categories, it is not all that we need to know about the syntax of sentences of our language.

Compare the following two sentences of our language

Pavarotti likes Sophia Loren.

Sophia Loren likes Pavarotti.

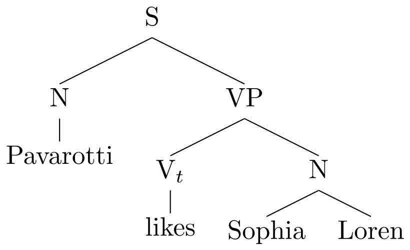
Both of these are grammatical sentences of our language; what's more, they are made up of just the same expressionss. Nonetheless, they have very different meanings. How can this be? What can explain the difference in meaning?

Intuitively the answer is obvious: they differ in meaning because, despite being made up of the same expressions, those expressions are combined in different ways in the two sentences. We need some way of representing this difference in 'mode of combination.'

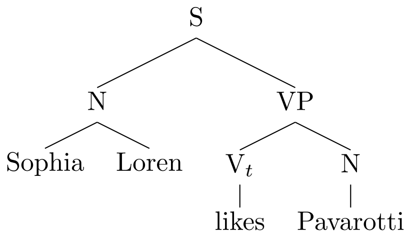
Given the specification of syntactic categories sketched above, the difference between the two sentences is that in the case of the first sentence, 'Sophia Loren' joins with the V_t 'likes' to form a verb phrase which then combines with the name 'Pavarotti' to form a sentence; whereas in the case of the second sentence, the roles are reversed.

What we need is a standard way of representing this difference. There are two, each of which is useful for different purposes.

The first is called a tree diagram, or phrase structure marker. The tree diagram of the first sentence would be



whereas the tree diagram for the second would be



Note that so far we have no account of *how* this difference could explain the difference in meaning between the two sentences; what we have is just a way of representing two different ways of obtaining a sentence from the same names and V_t.

We will spend a lot of time in this class using tree diagrams, and it will be useful to have some basic terminology for discussing them.

A *node* is a ‘location’ in the diagram, where a name for a syntactic category is.

> The *root node* is the top node, under which all the other nodes fall.
> A *parent* or (*mother*) node is one under which some other node falls.
> A *child* (or *daughter*) node is one which falls under some other node.
> A *branch node* is a parent node which is not the root node.
> A *leaf node* is one of the ‘bottom-level’ nodes — i.e., one which is not a parent node.
> The expressions which make up the sentence all fall under leaf nodes. They are called the *leaves* of the tree.

How many branch nodes do the above trees have?
How many child nodes?
How many leaf nodes?

How would you draw a tree diagram for:

> It is not the case that Pavarotti is cute.
> Pavarotti is boring and Sophia Loren is hungry.

The same structure can be represented using labelled brackets rather than a tree diagram. Using this notation, we can represent the structure of ‘Pavarotti likes Sophia Loren’ as follows:

> $[_{\text{S}}\ [_{\text{N}}\ \text{Pavarotti}]\ [_{\text{VP}}\ [_{\text{V}t}\ \text{likes}]\ [_{\text{N}}\ \text{Sophia Loren}\ ]]]$

In this notation, every node in the tree diagram corresponds to a set of brackets, with the subscript on the left bracket giving the syntactic category located at that node. What the tree diagram represents by placing some nodes under others is here represented by placing on set of brackets inside of another. So brackets for child nodes are located within the brackets of the corresponding parent nodes.

Anything which can be represented in a tree diagram can be represented using labeled bracketing; the choice between them is just a matter of convenience.

It might well seem at this point like this is an unnecessarily complex way of representing something very simple: namely that “Sophia Loren” is the direct object of the verb in the first sentence, whereas it is the subject of the second sentence. If the only sentences we ever had to deal with were this simple, this would be a good objection. This notation will earn its keep when we get to more complex constructions.