We left last time with this constraint on our grammars.

(1) Phrase Structure Grammars
A phrase structure grammar consists of a finite set of rules, each of which have the following form:
  a. Each rule rewrites exactly one symbol into a finite string of symbols.
     This is represented by “→”.
  b. The symbols are category labels.
Where category labels are both terminals and non-terminals.
The mapping between word types (i.e., syntactic category) and words we might represent with:

(2) a. D ⇒ the
    b. D ⇒ every
    c. N ⇒ cat
    d. N ⇒ hatred
    e. V ⇒ amuse
    f. V ⇒ push
    g. P ⇒ on
    h. P ⇒ because
    i. A ⇒ happy
    j. A ⇒ blue
    k. C ⇒ that
    l. C ⇒ if
Our rules so far are in (3).

(3) S → N V
    V → V P
    P → P D N
    C → C S
    α → (DisjD) α Disj α
Notice that this set of rules doesn’t obviously obey the definitions, as it’s not clear that the symbol “S” is a category label. We will have to return to this.

Because of (4), we’ll need to add the rules in (5).

(4) a. The dog jumped.
    b. The big dog jumped.
(5) a. N → D N
    b. N → A N
These rules are recursive, and that allows them to generate an infinite number of strings (and strings of infinite length). So, in addition to (4b), for instance, we can generate (6), in the way that (7) indicates.

(6) a. The big bad dog jumped.
    b. The big bad ugly dog jumped.
(7) S
    N V
    D N jumped
    the A N
    big A N
    bad dog
    S
    N V
    D N jumped
    the A N
    big A N
    bad A N
    ugly dog
Unfortunately, these rules also allow for the sentences in (8), which are ungrammatical.

(8) a. Bad the dog jumped.
    b. Every the dog jumped.
The problem is with the rule in (5a). We, first, need to make sure that this rule isn’t recursive and, second, that it doesn’t make the same kind of thing that can combine with an adjective. We need to make sure, then, that the thing that a D plus N combination makes isn’t another N. One way of doing that would be to change (5a) to (10), which will require that we redo the rule that makes sentences so that it looks like (11).

(10)  \( D \rightarrow D\ N \)
(11)  \( S \rightarrow D\ V \)

This will prevent (8) from being generated, but, because (27) is still recursive, it will allow things like (12).

(12)  The dog cat jumped.

We can see that same need if we look more closely at the second thing that makes a sentence. That thing we’ve called “V,” and we can see from sentences like (13) that it should allow it to be a V plus A sequence.

(13)  The dog is happy.

We could do that by adding to our rules the one in (14), which would allow (13) to have the parse in (15).

(14)  \( V \rightarrow V\ A \)
(15)  \( S \)

But because (14) is recursive, it will wrongly allow for things like (16).

(16)  The dog is happy big brown.

A popular hypothesis is that there is a difference between a string of words and an individual word that is at play in examples like these. The idea is that an individual word can be something that a string of words can’t be. We’ll make the thing that an individual word can be a “terminal,” and the thing that a string of words is a “non-terminal.” Non-terms are also often called “phrases,” and they are usually represented with a symbol that ends with “P” for phrase. For instance, we can
use “VP” to represent the phrase that ends an S. ("VP" stands for “verb phrase.”)
We’ll enrich the symbols our phrase structure rules manipulate to include both phrases and terminals.

(17) Phrase Structure Grammar
A phrase structure grammar consists of a finite set of rules, each of which have the following form:
   a. Each rule rewrites exactly one non-terminal into a finite string of terminals and non-terminals. This is represented by “→.”
   b. Terminals are category labels and non-terminals are phrases.

To bring our rules into conformity with this new definition, let’s change them to (18).

\[
\begin{align*}
S & \to DP \ VP \\
DP & \to D \ NP \\
NP & \to N \\
NP & \to AP \ NP \\
AP & \to A \\
CP & \to C \ S
\end{align*}
\]

\[
\begin{align*}
VP & \to V \\
VP & \to V \ DP \\
VP & \to VP \ PP \\
VP & \to VP \ CP \\
PP & \to P \ DP
\end{align*}
\]

I decided here to call the thing that a D+N makes a “determiner phrase” (DP). Another popular choice is to call this phrase a noun phrase (NP).

I’ve also changed the rule that makes P+D+N sequences (“prepositional phrases”) so that they have a phrase following the P, rather than just those three terminals. One way we can see that this is right is by remembering what we discovered about the rule that introduces or. That rule, recall, brings two things of the same kind together, with or positioned between them, and makes a larger thing of that kind. In our new system, we can write that rule with (19).

\[\alpha \to \alpha \ Disj \ \alpha, \text{ where } \alpha \text{ is any terminal or non-terminal.}\]

Sentences like (20), then, show us that the D+N part of a P+D+N sequence must be a phrase.

(20) The dog barked at the cat or the child.

The rules in (18) give (20) the parse in (21).

Notice that the words dog, cat and child are both terminals and non-terminals in this parse. That is, they are both an N and an NP. The same is true of the word barks in (22).

As you can see, one of the features of this new rule set is that every word will be both a terminal and part of a phrase. That phrase can contain other words, but it needn’t, as these examples illustrate.

Notice next how the use of the phrase/terminal distinction is used to control how many Adjectives can be inside a VP. The rule “VP → V AP” allows only one Adjective Phrase, and therefore Adjective, in any given VP. Compare that to the rule “VP → VP PP,” which, because it is recursive, allows for an indefinite number of PPs. This is illustrated by the parse in (23).
There is an important fact about phrases. They have a semantic property which makes them different from other strings of words. Phrases have a meaning that is the result of composing the meanings of the words they contain. That means that (a) they have a different meaning than merely the sum of the meanings of the words in them, and (b) that meaning comes from the meanings of the words inside. To see this, consider the two strings of words in (26).

(26)  

a. dog the happy  

b. the happy dog

The first string of words is just exactly that and nothing more. It is a list: three words and their associated meanings. But the second string of words has a meaning that is not just meanings of each of the words said in a row. “the happy dog” means something. It's meaning is a description of a particular thing – a thing that is a dog and is happy. We say about strings of words that have meanings like this that they are “compositional.” Phrases are compositional and if a string is compositional it is a phrase.

Let $\alpha$ be a collection of words $\{\beta_1, \ldots, \beta_n\}$, and let $[x]$ represent the meaning of $x$. If $[\alpha]$ is derivable from $[[\beta_1], \ldots, [[\beta_n]]$, but $[\alpha] \neq [[\beta_1], \ldots, [[\beta_n]]$, then $\alpha$'s meaning is compositional. The meaning of a collection of words is compositional if and only if it is a phrase.

This, it turns out, is the thing that plays a role in defining syntax. I've said that the syntax of a language is the set of laws that determine which strings of words is a sentence. To know what this means, we have to know what a "sentence" is. The key thing about sentences is that they are compositional. So, I can reframe the definition of syntax now with this in mind:

(28) Syntax is the set of laws that determine which strings of words are compositional.

Sentences are just a particular group of strings that are compositional. They are the ones that can stand alone as single utterances. The Law of Compositional, then, tells us that syntax is the set of laws that determine which strings of words are phrases. We still have to define "words," and, it will turn out, that there is a little bit more to syntax than this – but this is a very good beginning.

We can see how the law of compositionality, and the phrase structure rules we have, work by considering (29).

(29) She ate many crackers on every day.
In this sentence, *many crackers* provides a compositional meaning, but *crackers on* doesn’t. The rules we have do, in fact, make a phrase out of *many crackers* but not of *crackers on*. This will be one of our guiding criteria for choosing phrase structure rules: do they correctly capture just those strings which are compositional.

A special case of this arises when one string of words has two different ways of being put together. In such cases there will differrent parses, and consequently, different substrings that are compositional. Sometimes this will mean that there may be more than one meaning for a sentence. (30) is a case like that.

(30) She likes many big cats and dogs.

To know how to parse this sentence, we have to know how to position *and*. It turns out that* and* obeys the same rule that governs where *or* goes. This gives (30) the two parses in (31).

There are two ways our rules allow these words to be grouped and, indeed, two ways in which their meanings can be composed. On one, what the sentence says she likes are big cats and big dogs. On the other what the sentence says she likes are dogs and big cats.

If we set aside the rule that rewrites *S*, we can see a generalization in all the others. They all have one of the forms in (32).

A generalization about all of these rules is that they are all “projections,” in the sense defined by (33).

(33) A non-terminal *α* is a projection of *β* if *α* introduces *β* and:
   a. *β* is the unique terminal that *α* introduces, or
   b. *α* doesn’t introduce a terminal and *β* is the same as *α*.

We call the terminal that a phrase is recursively the projection of that phrase’s “head.” The head of a *VP* is a V, then; and the head of a *NP* is an N, and so on. Phrases that are projections are said to be “Endocentric.”

We might speculate that this pattern in the rules isn’t an accident. Indeed, this might be one of those things that is giving us a clue as to how the acquisition of syntax by a child is guided. Suppose, for instance, that as a child is putting together what the phrase structure rules for English are, her guesses are constrained so that she only postulates rules that are Endocentric. This would very seriously limit the kinds of rules she would entertain and, as a consequence, her eventual acquisition of the adult-like phrase structure rules might be enhanced. If the acquisition sequence is constrained in this way, we should see that all of the phrase structure rules of English – and indeed, every other language – are Endocentric. We might consider, then, limiting the class of grammars that syntax is limited to even further. They are not just drawn from the stock of context free rules, which operate over the terminal and non-terminal symbols we’ve seen. They are just those context free rules which are Endocentric. Here is our hypothesis then:

(34) Phrase Structure Grammars

The syntax of a language is a phrase structure grammar. A phrase structure grammar is a finite set of rules, each of which have the following form:

a. Each rule rewrites exactly one non-terminal into a finite string of terminals and non-terminals. This is represented by “→”.

b. Terminals are category labels and non-terminals are phrases.

c. Each rule is Endocentric.

But, of course, this leaves us with the problem that “*S → NP VP*” isn’t Endocentric.