CS 674/INFO 630: Advanced Language Technologies Fall 2007

Lecture 20 — November 8, 2007

Prof. Lillian Lee Scribes: *Cristian Danescu Niculescu-Mizil & Nam Nguyen & Myle Ott*

Analyzing Syntactic Structure

Up to now we have focused on the corpus as a whole and almost entirely ignored the structural details of each document. We claim that the sentence-level structure is very important when it comes to extracting the meaning of a text; the following two sentences are a clear example in which at least the order of the words is crucial:

```
Google bought YouTube.
YouTube bought Google.
```

For analyzing the structure of a sentence we will use constituent analysis, a hierarchical model introduced by Chomsky [1]. In this framework, sentences are modeled in the form of a parse tree (see Figure 1), where leaves are *lexical items*, or *terminals*, and internal nodes are *constituent labels.* The constituent labels form grammatical types from the lexical items, and lexical items form the sentence. For example, the first level of the tree in Figure 1 tells us that the analyzed sentence **S** is made out of a noun phrase **NP** and a verb phrase **VP**. To see this more clearly, let us introduce the following, more compact, notation:

$$[[\text{Police}]_{NP}\ [\text{put barricades around Upson}]_{VP}]_S$$

The constituent labels that are found immediately above the leaves correspond to parts-of-speech and serve as types for the lexical items; in our example **N**, **V**, and **Pr** correspond to the "noun," "verb," and "preposition" parts-of-speech.

One should take note of the *X-bar regularity*, which refers to the fact that for any *X-phrase*, **XP**, there exists a descendant of type **X** that is the *head* of the **XP**. In the example in Figure 1, the head of the first **NP** is the noun "Police" and the head of the **VP** is the verb "put."

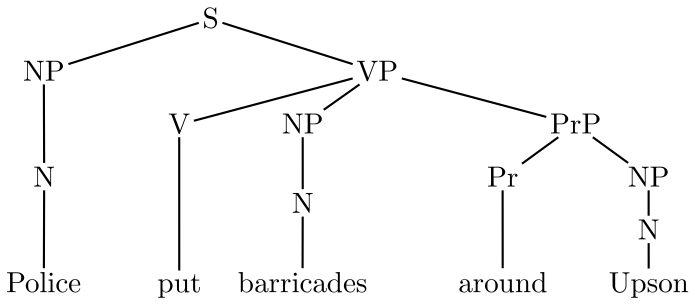


Figure 1: A parse tree for the sentence "Police put barricades around Upson." The constituent labels are: S=sentence, NP=noun phrase, VP=verb phrase, PrP=prepositional phrase, N=noun, Pr=preposition

It should also be mentioned that constituent analysis is just one of many types of hierarchical sentence-structure analysis. For example, one commonly used alternative to constituent analysis is dependency analysis ([6], [4]). In dependency analysis, relationships between words, rather than constituents, are labeled. Consequently, dependency analysis is insensitive to word ordering and is, therefore, often used for analyzing free-order languages. Figure 2 gives a dependency analysis analogous to the constituent analysis of Figure 1.

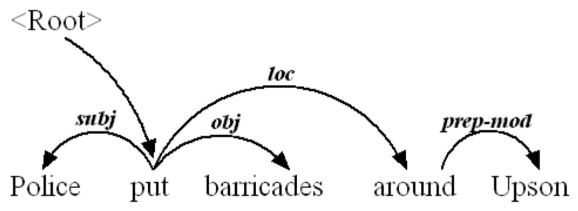


Figure 2: A dependency analysis for the sentence "Police put barricades around Upson."

Context Free Grammars

One important aspect of constituent analysis is distinguishing between correct and incorrect analyses. However, several ambiguities can arise with respect to syntactic structure and, therefore, one sentence can have more than one valid constituent analysis. To illustrate this, consider the following sentence:

I saw her duck [with a telescope]$_{PrP}$

The following ambiguities arise:

- PrP-attachment: the PrP "with a telescope" can modify any one of the three entities:

  I: I, using a telescope, saw her duck.

  her: I saw that she was holding a telescope while she ducked.

  duck: I saw a duck with a telescope, and the duck belonged to her.

- Part-of-speech (POS) ambiguity: the word "duck" can either be a noun or a verb.

It's important to note that not all combinations of the above choices are possible. For example, given that the word "duck" is a noun, the PrP cannot modify "her."

In addition to ambiguities in syntactic structure, there are also often semantic, or word sense, ambiguities. For example, in the above sentence, the word "saw" could have different meanings. Namely, it could refer to the act of sawing something (using a telescope). Alternatively, in the context of a poker game, she might have bet a duck and I might be "seeing" her bet with my own bet (a telescope).

It would be desirable to have a method to specify all and only valid constituent analyses; to do so we will start with employing Context Free Grammars (CFGs). For constituent analysis, a particular CFG, **G**, is given by:

- a finite set of lexical items (also called *terminals*)

- a finite set of constituent types (also called *non-terminals*)
- a finite set of one-level decompositions (also called *productions* or *rewrite rules*)
- a distinguished root type, S, corresponding to the sentence

For example, the CFG implied by Figure 1 is given by:

- **lexical items**: {police, put, barricades, around, Upson}
- **constituent types**: {S, NP, VP, PrP, N, V, Pr}
- **one-level decompositions**:

$$\begin{array}{lcll}
(1) & S & \rightarrow & NP \;\; VP \\
(2) & VP & \rightarrow & V \;\; NP \;\; PrP \\
(3) & PrP & \rightarrow & Pr \;\; NP \\
(4) & NP & \rightarrow & N \\
(5) & N & \rightarrow & police \\
(6) & N & \rightarrow & barricades \\
(7) & N & \rightarrow & Upson \\
(8) & V & \rightarrow & put \\
(9) & Pr & \rightarrow & around
\end{array}$$

- **root type**: S

A parse tree is valid with respect to **G** if and only if every branch is a decomposition given by **G** and all the leaves are lexical items. These grammars are called "context-free" because a decomposition of a constituent can be employed regardless of the context in which the constituent appears.

Problems with CFGs

In this section we look at some problems that arise from using a naïve CFG.

1) Proliferation of Categories

Consider the following three errors made by the CFG implied by Figure 1, which is given above. Notice that in each case, the most obvious solution is to create new categories to handle the erroneous cases. This is appropriately referred to as the "proliferation of categories" problem.

a. Selectional Error

Let us illustrate the error by way of example:

$$\# \, [[\text{Upson}]_N \, [\text{put}]_V \, [\text{barricades}]_N \, [\text{around}]_{Pr} \, [\text{police}]_N]_S$$

While the above sentence is "semantically doubtful" (indicated by the leading '#'), our CFG would accept it. This is referred to as a "selectional error," and refers to the need for semantic information in type labels. In this case, the "selection error" is that Upson[1] is inanimate and, thus, cannot actively *put* anything around anything else.

b. Case Mismatch

Let us again illustrate by example:

$$* \, [[\text{Police}]_N \, [\text{put}]_V \, [\text{they}]_N \, [\text{around}]_{Pr} \, [\text{Upson}]_N]_S$$

This sentence is syntactically incorrect (indicated by the leading '∗') because, while both "they" and "them" are nouns, the former is clearly incorrect in this sentence. The problem here is that "they" and "them" are different *types* of nouns, i.e. they are not interchangeable because they have different cases.[2] Clearly this must be accounted for in our grammar.

c. Agreement Mismatch

We again illustrate through example:

$$* \, [[\text{Police}]_N \, [\text{enforces}]_V \, [\text{the}]_{Art} \, [\text{rules}]_N]_S$$

The problem exhibited by the above sentence is an agreement mismatch between "Police" (3^{rd} person plural) and "enforces" (3^{rd} person singular). A possible correction is to replace "enforces" with "enforced," but in any case it is obvious that agreement information is also necessary in our type labels.

2) Long Distance Dependencies

Let us now consider some problems that arise because of "long distance" lexical dependencies. Two situations in which these dependencies occur are question inversion and wh-movement. For example:

$$\text{Police} \, [[\text{put}]_V \, [\text{barricades}]_N \, [\text{around Upson}]_{PrP}]_{VP}$$

might become,

$$\text{What did police} \, [[\text{put}]_V \, [\text{around Upson}]_{PrP}]_{VP}$$

[1]Actually, Upson Hall is named after someone named Upson, but we are ignoring this here, and assuming 'Upson' refers to the building.

[2]In other languages, it's not just pronouns that are marked explicitly for case.

The naïve rule for verb phrases in this case would be:

$$\begin{aligned} VP &\rightarrow V \;\; PrP \\ VP &\rightarrow V \;\; NP \;\; PrP \end{aligned}$$

However, this would allow the sentence “Police put around Upson” to be generated, which is clearly incorrect.

There are more problems regarding question inversion, though. Notice in the following sentences that while the verb phrases are correct, the question forms are not.

Where did police $[\text{put around Upson}]_{VP}$

What did police $[\text{put barricades around Upson}]_{VP}$

The problem with all of these examples is that it is not obvious how to model dependencies between terms using a CFG when terms change positions. One possible account will be discussed in the following lecture.

Question

We now explore a possible solution to some of the dependency problems discussed above. Consider the following sentences:

1. `Police put barricades around Upson.`
2. `What did police put around Upson?`

Now consider a reasonable grammar used to generate these sentences:

$$\begin{array}{rccl}
(1) & S & \rightarrow & NP \;\; VP \\
(2) & S & \rightarrow & NP_{wh} \;\; Aux \;\; N \;\; VP \\
(3) & NP & \rightarrow & N \\
(4) & VP & \rightarrow & V \;\; NP \;\; PrP \\
(5) & VP & \rightarrow & V \;\; PrP \\
(6) & PrP & \rightarrow & Pr \;\; NP \\
 & & & \\
(7) & NP_{wh} & \rightarrow & what \\
(8) & Aux & \rightarrow & did \\
(9) & N & \rightarrow & police \\
(10) & N & \rightarrow & barricades \\
(11) & N & \rightarrow & Upson \\
(12) & V & \rightarrow & put \\
(13) & Pr & \rightarrow & around
\end{array}$$

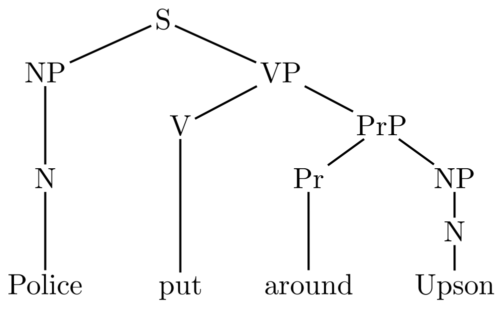
Note that the following, incorrect, sentences can also be generated:

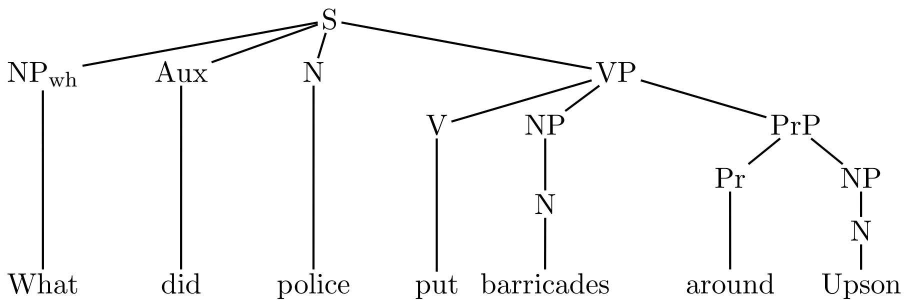
a) `Police put around Upson.`

b) `What did police put barricades around Upson?`

Show how these can be generated by giving the respective parse trees.

Answer





Question

It is obvious that a misuse of rule (5) is to blame for sentence (a). Without adding any new rules, propose a substitute rule for (5) such that the resulting grammar no longer generates sentence (a), but still generates sentences 1 and 2 (show this by giving the parse trees for each sentence). Note that the resulting grammar does not necessarily have to be context-free.

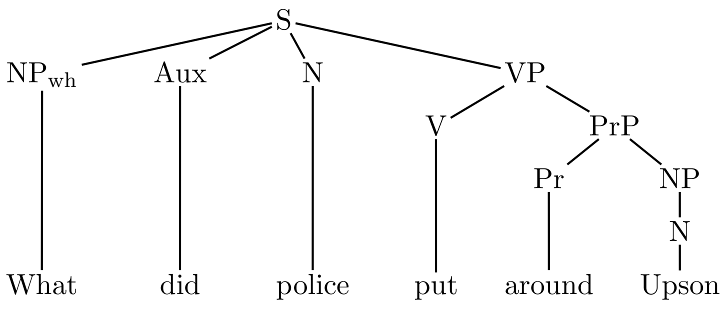
Answer

To achieve the desired result without adding any new rules, we will make use of a context-sensitive formalism[3] to introduce context information into rule (5). We get:

$$(5') \quad NP_{wh} \;\; Aux \;\; N \;\; VP \;\; \rightarrow NP_{wh} \;\; Aux \;\; N \;\; V \;\; PrP$$

The added context-constraint in (5′) ensures that the rule is only applied in cases where the sentence is a "wh-question."

The parse tree for sentence 1 is given by Figure 1 and the parse tree for sentence 2 is given below:



Question

Even after making the change given by the previous question, our grammar still incorrectly generates sentence (b). Add context-sensitivity to rule (4) in order to obtain a new grammar which

[3]We refer to the class of Context-Sensitive Grammars (CSGs) which allow one-level decompositions of the form: $\alpha A \beta \rightarrow \alpha\gamma\beta$ where A is a constituent type and α, β and γ are strings of constituent types and lexical items; except when $A = S$, γ must be nonempty.

will neither generate sentence (a) nor sentence (b), but still generates sentences 1 and 2 (as before, show this by giving the parse trees for each sentence).

Answer

We can replace production (4) with:

$$(4') \quad NP \;\; VP \;\; \rightarrow NP \;\; V \;\; NP \;\; PrP$$

In this case, the parse tree for sentence 1 is again given by Figure 1 and the parse tree for sentence 2 is the same as in the previous exercise.

Question

It is widely conjectured that, unlike context-sensitive grammars, context-free grammars are not powerful enough to represent natural language [2] [5]. If CFGs are not sufficiently powerful to represent natural language though, it is not obvious why they should be relevant to NLP research. Furthermore, if context-sensitive grammars *are* powerful enough, it is not clear why there is a reluctance to use context-sensitive formalisms. Explain why CFGs *are* relevant to NLP research and why context-sensitive grammars are often avoided.

Answer

The most obvious reason that context-sensitive grammars might be unattractive to NLP research is because of efficiency concerns. Namely, it has been shown that the context-sensitive parsing problem is PSPACE-Complete. Thus, while context-sensitive grammars provide a great deal of expressive power, efficient parsers are known to exist only for context-*free* grammars.

However, while context-free grammars might lack sufficient power to express all of natural language, it certainly seems that a large portion of natural language can be expressed using CFGs. In fact, there exist constructed spoken languages for which CFGs *are* sufficiently powerful (e.g. Lojban). As it turns out, many of the commonly employed formalisms lie between context-free and fully context-sensitive grammars, i.e. formalisms that are more expressive than CFGs but easier to work with than CSGs. One example is the tree adjoining grammars [3] which we will treat in a future lecture.

References

[1] Chomsky, N. "Three models for the description of language." In I.R.E. Transactions on Information Theory 2, pp. 113-124 (1956)

[2] Culy, C. "The complexity of the vocabulary of Bambara." Linguistics and Philosophy 8:345-351 (1985)

[3] Joshi, A. K., L. S. Levy, M. Takahashi. “Tree adjunct grammar.” Journal of Computer and System Sciences 10 (1): 136-163 (1975)

[4] Mel’čuk, I. “Studies in Dependency Syntax.” Ann Arbor, MI: Karoma (1979)

[5] Shieber, S. “Evidence against the context-freeness of natural language.” Linguistics and Philosophy 8:333-343 (1985)

[6] Tesnière, L. “Éléments de syntaxe structurale.” Editions Klincksieck (1956)