CS 6740/INFO 6300 **Spring 2010**

Lecture 17 – April 15

Lecturer: Prof Lillian Lee *Scribes: Jerzy Hausknecht & Kent Sutherland*

Using & Augmenting Context-Free Grammars for Sequential Structure

1 Motivation for Context-Free Grammars

A context-free grammar (CFG) is a simple but flexible way to describe syntactic structure. In using CFGs, it is important to keep in mind that the ultimate goal is to recover the underlying structure of natural language sentences. Although we will encounter various weaknesses of CFGs, they are still extremely useful due to the existence of efficient algorithms for recovering hidden structure with respect to a fixed CFG (such as the CYK algorithm). Their flaws can be addressed in a myriad of ways, but we will see that fixes complicate the initially simple formalism.

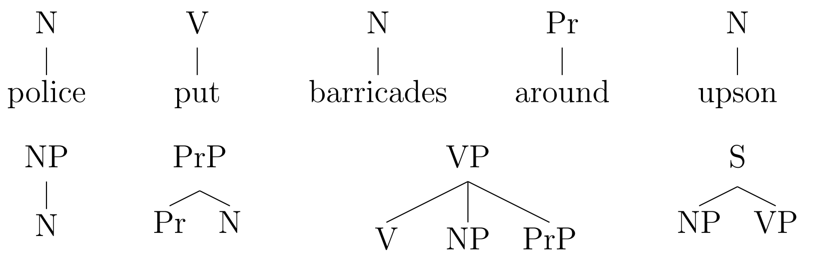
2 Context-Free Grammar Overview

A CFG is defined by:

- A finite set of vocabulary items, also called lexical items or terminals. These are atomic, and therefore not decomposable. By convention, they are indicated by lowercase letters.
- A finite set of constituent types or categories, which are decomposable into constituent types or lexical items. By convention, they are indicated by capital letters.
- A finite set of production rules, which define how the constituent types can be decomposed. These rules correspond to one-level branches in a tree.
- A single symbol, which must be a valid constituent type, for the root of any parse tree considered to be complete.

2.1 Parse Example

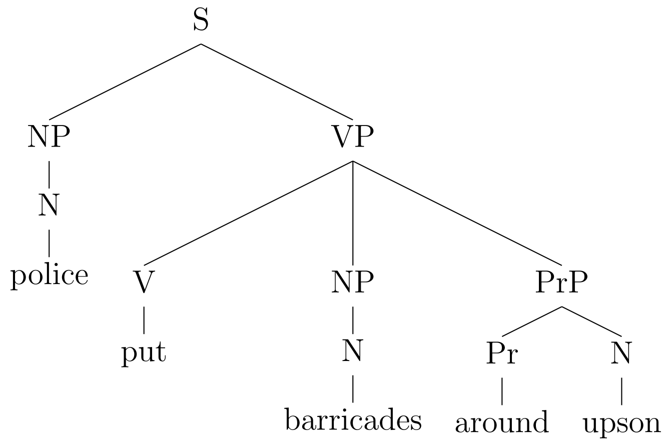
Consider the following context-free grammar:



S = sentence
NP = noun phrase
VP = verb phrase
PrP = prepositional phrase
N = noun
V = verb
Pr = preposition

In the above grammar, the lexical items are the English words in the leaves of the first row of production rules. The constituent types are enumerated in the right column. The second row of production rules defines how constituent types can be further decomposed into more constituent types. For the purposes of this grammar, let S be the single valid root symbol.

Using the above grammar, the following is a parse of the sentence "police put barricades around upson".



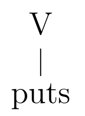
The fringe, or set of leaves, is exactly the original sentence. Each leaf must be a lexical item. Additionally, all nodes in the tree are either valid constituent types or lexical items. Furthermore, the tree is rooted by the distinguished category S, which represents the entire sentence. Lastly, each decomposition in the tree of a constituent type corresponds to a valid branch according to the defined grammar.

3 Shortcomings of Context-Free Grammars

Although the grammar we have just described appears to exemplify a simple and elegant way to describe sentences, there are a number of problems with it.

3.1 Agreement Mismatch

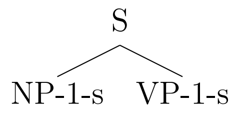
Consider what happens when we modify the above context-free grammar by adding a new, and seemingly acceptable, rule, as a way to start allowing sentences like "She puts silverware on the table."



Unfortunately, this allows us to generate the following sentence: “police puts barricades around upson.” Clearly, this sentence is syntactically ill-formed. More specifically, this is an agreement mismatch error, as the tense of the subject and verb don’t agree. “police” is a third-person plural noun, while “puts” is a third-person singular verb.

3.1.1 Possible Solution

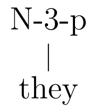
As a workaround, we can track the grammatical person and plurality of each word. This necessitates imposing a constraint that forces all entities in a “local” branch to agree in grammatical person and plurality. Instead of V representing a verb, we use V-1-s to represent a first-person singular verb, V-1-p to represent a first-person plural verb, V-3-s to represent a third-person plural verb, and so on. The same changes are necessary for other parts of speech in our grammar, such as nouns, noun phrases, and verb phrases. The following is an example of the type of new decomposition rules required by these changes.



Although this change seems to address the agreement mismatch, it does result in a constant-factor increase in the complexity of our grammar. Such a solution could be considered satisfactory if this were the only problem, but there are other issues as well.

3.2 Case Mismatch

Let’s make another change to the context-free grammar, adding the following new rule.



This allows us to generate the new sentence “they put barricades around upson.” Unfortunately, it also gives us the following sentence: “police put they around upson”. This sentence is ill-formed, as “they” should instead be “them”. This is an example of a case mismatch. “them” is accusative, which correctly indicates a direct object. “they” is nominative, a subject.

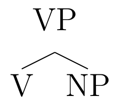
3.2.1 Possible Solution

We can employ a workaround similar to the one used to address agreement mismatches. In English, this doesn’t dramatically increase the complexity of the resulting grammar, as only pronouns explicitly indicate case. However, other languages are considerably more complex in this regard. Historically, Indo-European languages had eight cases [Wikipedia]. Furthermore, other languages have additional categories that aren’t present in English, such as gender. The result is a proliferation of categories that results in an extremely complex grammar.

3.3 Other Problems

3.3.1 Sub-categorization Error

Consider the following new rule:



This rule correctly allows sentences like “police saw criminals” (assuming the addition of appropriate extra rules) but incorrectly allows the sentence “police put barricades”. This is a sub-categorization error, where the expected type and cardinality of arguments do not match the input sentence. In this example, the verb “put” requires a place to put the barricades. As before, we can encode these requirements in the parts of speech. For example, V-1-s-1-NP-arg would correspond to a first-person singular verb that expects a single noun-phrase argument as a direct object. This adds yet another layer of complexity, and introduces an undesirable aspect: it duplicates structural information in both the tree structure and the category names of the grammar.

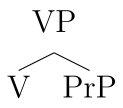
3.3.2 Selectional Violation

Consider the sentence “barricades put upson around police”. Although this sentence is syntactically valid according to our grammar, it is semantic nonsense. This particular error is called a selectional violation. The verb “puts” requires that its subject be animate (something that can put things). Additionally, its object must be something that can be put somewhere (a “concrete” noun). The previous sentence violates the first of these constraints, as “barricades” are not animate. Selectional violation is a more serious problem in terms of category proliferation, as we no longer have a tightly closed set of features that can be accounted for. While the number of features remains finite, the number of new categories goes far beyond a small constant factor increase.

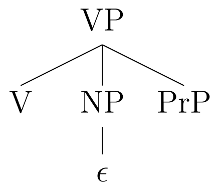
3.4 Beyond Category Proliferation

Another weakness of CFGs is in enforcing long-distance dependencies in the parse tree. An example of such a dependency is question inversion. Consider the following sentence: “what did police put around upson?”. There is clearly a dependency between the words “what” and “put,” even though they are not adjacent. For example, the sentence “where did police put around upson?” is invalid, as “where” doesn’t agree with put’s requirement for an object that can be put as its direct object. Also, “what did police put barricades around upson?” is invalid, as put’s direct object is both “what” and “barricades”, which is syntactically invalid. Given these examples, we can see that there is a dependency between the position after "put" (which would normally contain the direct object) and the wh-pronoun at the beginning of the question.

The inability to generate the sentence "what did police put around upson?" could be solved superficially by adding the following rule, which would remove the requirement that the verb has a direct object directly adjacent:



However, this rule would also allow the following phrase: "police put around upson". This sentence is invalid English, and demonstrates the need for representing the position of the direct object. Therefore, consider the following structure:



The new lexical item ϵ, sometimes called a "trace," is the phonological equivalent to null. In order to be useful, this null symbol needs to be somehow "linked" through the rest of the tree to the wh-pronoun at the beginning of the question. Unfortunately, this isn't possible using our current context-free grammar.

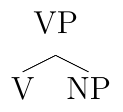
4 Feature-Based CFGs with Unification Constraints

As we have seen, CFGs have two major problems: long-distance dependencies and category proliferation. A potential solution to both these problems is the use of feature-based CFGs with unification constraints. The basic idea is to encode the language features of lexical items in the lexicon as much as possible, thus minimizing category proliferation within the constituent types. Unification constraints are then used to enforce the requirements of a decomposition. We use an attribute-value matrix (AVM) to store the features, as shown in the following lexical entry for "notify." For simplicity, assume that this definition of the word takes only one argument and is third person plural. This means we are ignoring the fact that for the sentence "police must notify parents," "notify" is modified by the modal verb "must" and isn't third person plural.

$$\begin{bmatrix} \text{CAT} & \text{V} \\ \text{ORTH} & \text{notify} \\ \text{HEAD} & \begin{bmatrix} \text{SUBCAT} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{RESTRICT} & +\text{animate} \end{bmatrix} \end{bmatrix} \\ \text{FIRST} & \begin{bmatrix} \text{CAT} & \text{NP} \\ \text{HEAD} & \begin{bmatrix} \text{RESTRICT} & +\text{animate} \end{bmatrix} \end{bmatrix} \\ \text{SECOND} & \text{end} \end{bmatrix} \\ \text{AGREEMENT} & \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{plural} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Note that the feature names are on the left side of the vectors, and the values are on the right. CAT indicates the category that the lexical item belongs to – a verb in this instance. ORTH stands for orthography or spelling. HEAD contains features that are passed to the constituent type that is the lexical item's parent in the parse tree through unification constraints. SUBCAT, under HEAD, contains requirements that the lexical item imposes on the other parts of the sentence. RESTRICT stands for selectional restrictions on features, which prevents selectional violations from occurring. Additionally, the fact that SECOND under SUBCAT has the value "end" indicates that the verb only expects one argument. In this example, "notify" requires that the subject of the sentence be animate, that its first argument be a noun phrase that is also animate, and that it have no second argument. Lastly, AGREEMENT indicates the grammatical person of the verb.

Consider the following branch for a verb with one argument, as well as a small set of unification constraints. Notice how we have gone back to using "non-decorated" categories for the constituent types.



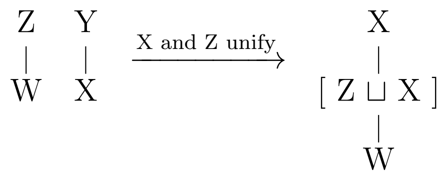
<VP HEAD> = <V HEAD>
<V HEAD SUBCAT FIRST CAT> = NP
<V HEAD SUBCAT SECOND> = end
<NP HEAD CASE> = accusative

The first constraint is how information is passed up the tree. Enforcing consistency across HEAD features ensures XP regularity; namely that the important features of a constituent are inherited from its head. For example, the verb phrase "put barricades that she regularly paints in a special place" is third person plural, from the verb "put," rather than third person singular from "paints." The second constraint enforces that the verb expects a noun phrase as its first argument, and the third constraint enforces that the verb expects only one argument. Finally, the fourth constraint enforces that the case of the direct object is correct. Obviously, these constraints are incomplete, but they form the basis of our new grammar.

The unification constraints are indicated using the equals sign, which indicates that the two attribute-value matrices can unify with each other. When two feature structures are unified, the result is the most specific generalization encompassing the two. For example, consider the following two AVMs: [PERSON 1] and [PERSON 3]. In this case, no unification is possible. However, consider the following different AVMs: [PERSON 1] and [NUMBER singular]. The resulting unification is:

$$\begin{bmatrix} \text{PERSON} & 1 \\ \text{NUMBER} & \text{singular} \end{bmatrix}$$

Unification constraints also affect how branch addition works. Using a standard context-free grammar, two branches of a tree can be combined only if the constituent types match. With unification constraints, branch addition works as follows, where W, X, Y, and Z stand for feature structures, and aren't constituent types themselves:



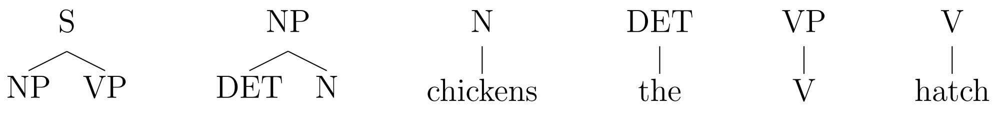
Using this formalism, we can address the other problem of long-distance dependencies. Unification constraints allow us to correctly represent the sentence “whom did police notify?” by propagating the noun-phrase ϵ up the tree through a feature that we will call GAPINFO. Future lectures will explore this concept in more detail.

5 Questions

5.1 Syntactic Errors & Tense

The following question explores some of other ways in which problems resulting in category proliferation occur.

Consider the sentence, “the chickens hatch,” generated from the following rules:



This sentence is in the present tense, but now we want to express the same thought in a different tense.

a) Pretend that this event occurred in the past. What additional rule could be added to the grammar to form the sentence “the chickens hatched?”

b) Now the action will occur in the future: “the chickens will hatch.” What additional rules will be needed to generate this sentence? For now, assume “will” is a standard verb.

c) Consider the grammar that we have after adding new rules for the first two parts. Does this grammar generate any problematic sentences? Indicate the type of error that occurs for each invalid sentence.

d) Rather than treating “will” as a normal verb, treat it as an auxiliary verb (a different part of speech). Does this fix any errors from the previous question? Do any problems remain?

5.2 Punctuation in Grammar

Punctuation plays an important role in understanding language. A now-famous example of this is the title of the 2003 book “Eats, Shoots & Leaves: The Zero Tolerance Approach to Punctuation,” by Lynne Truss:

A panda walks into a café. He orders a sandwich, eats it, then draws a gun and proceeds to fire it at the other patrons.

"Why?" asks the confused, surviving waiter amidst the carnage, as the panda makes towards the exit. The panda produces a badly punctuated wildlife manual and tosses it over his shoulder.

"Well, I'm a panda," he says, at the door. "Look it up."

The waiter turns to the relevant entry in the manual and, sure enough, finds an explanation. "Panda. Large black-and-white bear-like mammal, native to China. Eats, shoots, and leaves."

While a panda that eats shoots and leaves is nothing noteworthy, a panda that eats, shoots, and leaves is something generally not encountered in day-to-day life. Other ambiguities can also arise from punctuation.

This clear difference between "eats shoots and leaves" and "eats, shoots, and leaves" is that the appearance of commas changes the meaning of "shoots" and "leaves." Without commas, the two words are easily construed as nouns, but with commas the words become verbs. This occurs because the appearance of commas turns the words into a list, where the items in the list must be of the same constituent type.

a) Assume we are using a context-free grammar that uses parts of speech as its constituent types. What additional rules would be necessary to encode the following sentence fragment: "runs, bikes, and swims".

b) As we have seen, feature-based context-free grammars present a means of avoiding category proliferation. Since the new rules in the previous part would still presumably suffer from syntactic errors, we would like to encode the list requirement in an attribute value matrix. What might the basic structure of such a matrix look like?

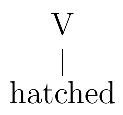
5.3 Practicality of CFGs

(Mental exercise, no answer provided) Much of our discussion of context-free grammars has focused on the challenges of preventing the generation of syntactically invalid sentences. However, if the ultimate goal is to analyze natural language, then it must be possible to allow for structural analysis of grammatically incorrect sentences. Does this invalidate our previous attempts to prevent the generation of agreement errors, sub-categorization errors, and selectional violations? How can we allow for the parsing of naturally-occurring grammatically incorrect sentences while still retaining meaningful sentential structure?

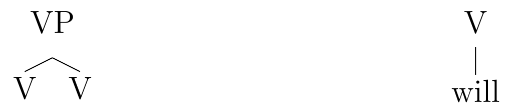
6 Answers

6.1 Syntactical Errors & Tense

a) The following rule will allow this sentence to be constructed using the grammar:



b) Two new rules are necessary:

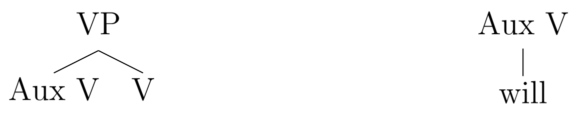


c) Considering these three new rules, there are a number of problematic sentences.

1) "the chickens will hatched"
2) "the chickens will will"
3) "the chickens hatched will"
4) "the chickens hatch will"
5) "the chickens hatched hatched"
6) "the chickens hatched hatch"
7) "the chickens hatch hatch"

The first and second sentences exhibit sub-categorization errors, where the second verb's tense does not match what the first verb is expecting. The remaining sentences are also sub-categorization errors, due to the first verb not expecting a second verb to be directly adjacent.

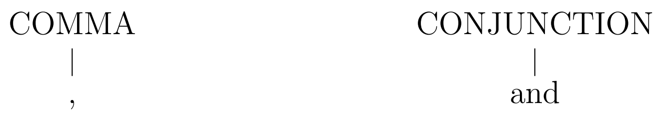
d) We can eliminate some problems by defining the use of "will" more narrowly.

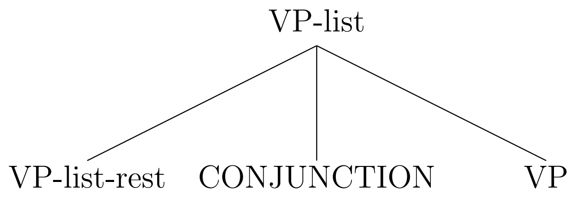


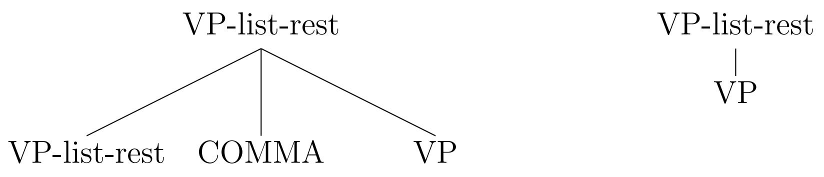
Unfortunately, there is still a potential sub-categorization error: "the chickens will hatched" could be generated from these rules.

6.2 Punctuation in Grammar

a) Recursive constituent types could be added to encode the given fragment. This ensures that the part of speech of all items in a list match each other:



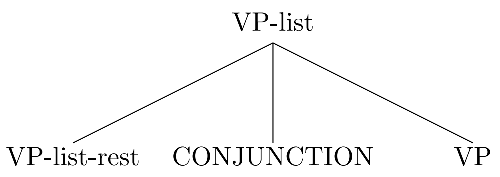




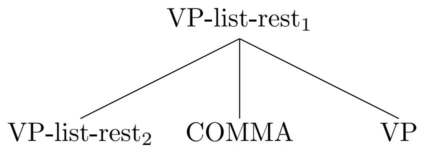
b) One solution would be to encode lists as a feature, similar to how we encoded the number of arguments earlier. Rather than using FIRST and SECOND features, we can treat the list of words similar to a linked list through a NEXT feature.

$$\begin{bmatrix} \text{CAT} & \text{VP-list-rest} \\ \text{ORTH} & \ldots \\ \text{HEAD} & \begin{bmatrix} \text{SUBCAT} & \begin{bmatrix} \text{NEXT} & \begin{bmatrix} \text{CAT} & \text{VP} \\ \text{HEAD} & [\ \ldots \quad \ldots\ ] \end{bmatrix} \end{bmatrix} \\ \ldots & [\ \ldots \quad \ldots\ ] \end{bmatrix} \end{bmatrix}$$

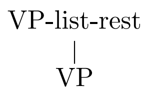
We also need new unification constraints:



<VP-list HEAD> = <VP-list-rest HEAD>
<VP-list HEAD> = <VP HEAD>
<VP-list-rest HEAD SUBCAT NEXT HEAD> = <VP HEAD>



<VP-list-rest$_1$ HEAD> = <VP-list-rest$_2$ HEAD>
<VP-list-rest$_1$ HEAD> = <VP HEAD>
<VP-list-rest$_2$ HEAD SUBCAT NEXT HEAD> = <VP HEAD>



<VP-list-rest HEAD> = <VP HEAD>

Note that we have left out any unification rules for conjunctions. If we were to actually implement such a system, we would also need rules for conjunctions. However, handling unification of the HEAD SUBCAT feature between two words joined by a conjunction can be problematic. For example, “eats, drinks beer, and leaves” contains two intransitive verbs, and one transitive verb. If these requirements are indicated as the HEAD features, it would not be possible to unify the given sentence. Obviously some requirements will need to be enforced, as otherwise invalid sentences could be generated. This currently an open area of research.

7 References

1. Wikipedia contributors. "CYK algorithm." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 14 Apr. 2010. Web. 21 Apr. 2010.

2. Wikipedia contributors. "Grammatical case." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 18 Apr. 2010. Web. 21 Apr. 2010.

3. Truss, Lynne. "Eats, Shoots & Leaves: The Zero Tolerance Approach to Punctuation." Profile Books, UK. 2003.