Natural Language Processing

Lecture 14—10/13/2015

Jim Martin

Today

Moving from words to larger units of analysis

Syntax and Grammars

- Context-free grammars
- Grammars for English
- Treebanks
- Dependency grammars

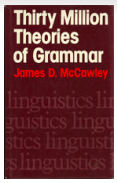
- Moving on to Chapters 12 and 13

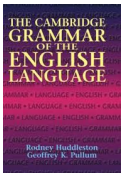
Syntax

- By syntax, we have in mind the kind of implicit knowledge of your native language that you had mastered by the time you were 3 years old without any explicit instruction
- Not the kind of stuff you were later taught about grammar in "grammar" school

Syntax in Linguistics

- Phrase-structure grammars, transformational syntax, X-bar theory, principles and parameters, government and binding, GPSG, HPSG, LFG, relational grammar, minimalism…
- Reference grammars: less focus on theory and more on capturing the facts about specific languages





Syntax

- Why do we care about syntax?
- Grammars (and parsing) are key components in many practical applications
  - Grammar checkers
  - Dialogue management
  - Question answering
  - Information extraction
  - Machine translation

Syntax

- Key notions that we will cover
  - Constituency
    - And ordering
  - Grammatical relations and dependency
    - Heads, agreement, grammatical function
- Key formalisms
  - Context-free grammars
  - Dependency grammars
- Resources
  - Treebanks

Constituency

- The basic idea here is that groups of words within utterances can be shown to *act as single units*
- And in a given language, these units form coherent classes that can be be shown to behave in similar ways
  - With respect to their internal structure
  - And with respect to other units in the language

Constituency

- Internal structure
  - We can ascribe an internal structure to the class
- External behavior
  - We can talk about the constituents that this one commonly associates with (follows, precedes or relates to)
    - For example, we might say that in English noun phrases can precede verbs

Constituency

- For example, it makes sense to the say that the following are all ***noun phrases*** in English...

Harry the Horse	a high-class spot such as Mindy's
the Broadway coppers	the reason he comes into the Hot Box
they	three parties from Brooklyn

- Why? One piece of evidence is that they can all precede verbs.
  - That's what I mean by external evidence

Grammars and Constituency

- Of course, there's nothing easy or obvious about how we come up with right set of constituents and the rules that govern how they combine...
- That's why there are so many different theories of grammar and competing analyses of the same data.
- The approach to grammar, and the analyses, adopted here are very generic (and don't correspond to any modern, or even interesting, linguistic theory of grammar).

Context-Free Grammars

- Context-free grammars (CFGs)
  - Also known as
    - Phrase structure grammars
    - Backus-Naur form
- Consist of
  - Rules
  - Terminals
  - Non-terminals

Context-Free Grammars

- Terminals
  - Take these to be words (for now)
- Non-Terminals
  - The constituents in a language
    - Like noun phrase, verb phrase and sentence
- Rules
  - Rules consist of a single non-terminal on the left and any number of terminals and non-terminals on the right.

Some NP Rules

- Here are some rules for our noun phrases

$$\begin{aligned} NP &\rightarrow Det\ Nominal \\ NP &\rightarrow ProperNoun \\ Nominal &\rightarrow Noun \mid Nominal\ Noun \end{aligned}$$

- Together, these describe two kinds of NPs.
  - One that consists of a determiner followed by a nominal
  - And another that says that proper names are NPs.
  - The third rule illustrates two things
    - An explicit disjunction
      - Two kinds of nominals
    - A recursive definition
      - Same non-terminal on the right and left-side of the rule

L0 Grammar

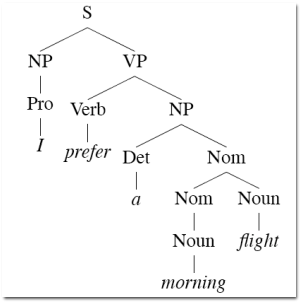
Grammar Rules			Examples
S	$\rightarrow$	$NP\ VP$	I + want a morning flight
NP	$\rightarrow$	$Pronoun$	I
	$\mid$	$Proper\text{-}Noun$	Los Angeles
	$\mid$	$Det\ Nominal$	a + flight
$Nominal$	$\rightarrow$	$Nominal\ Noun$	morning + flight
	$\mid$	$Noun$	flights
VP	$\rightarrow$	$Verb$	do
	$\mid$	$Verb\ NP$	want + a flight
	$\mid$	$Verb\ NP\ PP$	leave + Boston + in the morning
	$\mid$	$Verb\ PP$	leaving + on Thursday
PP	$\rightarrow$	$Preposition\ NP$	from + Los Angeles

Generativity

- As with finite-state machines and HMMs, you can view these rules as either analysis or synthesis engines
  - Generate strings in the language
  - Reject strings not in the language
  - Assign structures (trees) to strings in the language

Derivations

- A *derivation* is a sequence of rules applied to a string that *accounts* for that string
  - Covers all the elements in the string
  - Covers only the elements in the string



Definition

- Formally, a CFG consists of

N	a set of **non-terminal symbols** (or **variables**)
Σ	a set of **terminal symbols** (disjoint from N)
R	a set of **rules** or productions, each of the form $A \rightarrow \beta$, where A is a non-terminal, β is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$
S	a designated **start symbol**

Parsing

- Parsing is the process of taking a string and a grammar and returning parse tree(s) for that string
- It is analogous to running a finite-state transducer with a tape
  - It's just more powerful
    - This means that there are languages we can capture with CFGs that we can't capture with finite-state methods
    - More on this when we get to Ch. 13.

Example

An English Grammar Fragment

- Sentences
- Noun phrases
  - Agreement
- Verb phrases
  - Subcategorization

Sentence Types

- Declaratives: *A plane left.*
  S → NP VP
- Imperatives: *Leave!*
  S → VP
- Yes-No Questions: *Did the plane leave?*
  S → Aux NP VP
- WH Questions: *When did the plane leave?*
  S → WH-NP Aux NP VP

Noun Phrases

- Let's consider the following rule in more detail...

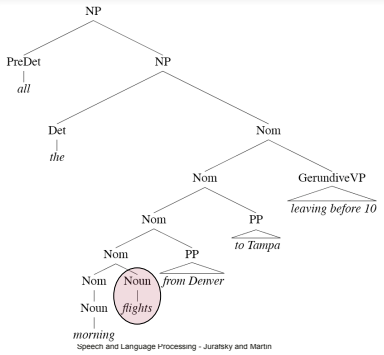
  NP → Det Nominal

- Most of the complexity of English noun phrases is hidden inside this one rule.
- Consider the derivation for the following example
  - *All the morning flights from Denver to Tampa leaving before 10...*

NP Structure

- Clearly this NP is really about "*flights*". That's the central organizing element (noun) in this NP.
  - Let's call that word the *head*.
  - All the other words in the NP are in some sense *dependent* on the head
- We can dissect this kind of NP into
  - the stuff that comes before the head
  - the head
  - the stuff that comes after it.

Noun Phrases



Determiners

- Noun phrases can consist of determiners followed by a nominal

  NP → Det Nominal

- Determiners can be
  - Simple lexical items: *the, this, a, an*, etc.
    - A car
  - Or simple possessives
    - John's car
  - Or complex recursive versions of possessives
    - John's sister's husband's son's car

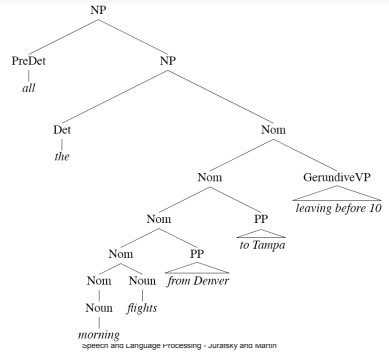
Nominals

- Contain the head and any pre- and post-modifiers of the head.
  - Pre-
    - Quantifiers, cardinals, ordinals...
      - *Three cars*
    - Adjectives
      - *large cars*

Postmodifiers

- Three kinds
  - Prepositional phrases
    - *From Seattle*
  - Non-finite clauses
    - *Arriving before noon*
  - Relative clauses
    - *That serve breakfast*
- Same general (recursive) rules to handle these
  - *Nominal → Nominal PP*
  - *Nominal → Nominal GerundVP*
  - *Nominal → Nominal RelClause*

Noun Phrases



Verb Phrases

- English *VP*s consist of a verb (the head) along with 0 or more *following* constituents which we'll call *arguments*.

VP	→	*Verb*	disappear
VP	→	*Verb NP*	prefer a morning flight
VP	→	*Verb NP PP*	leave Boston in the morning
VP	→	*Verb PP*	leaving on Thursday

Subcategorization

- Even though there are many valid VP rules in English, not all verbs are allowed to participate in all those VP rules.
- We can *subcategorize* the verbs in a language according to the sets of VP rules that they participate in.
- This is just an elaboration on the traditional notion of transitive/intransitive.
- Modern grammars have many such classes

Subcategorization

- Sneeze: John sneezed
- Find: Please find [a flight to NY]$_{NP}$
- Give: Give [me]$_{NP}$[a cheaper fare]$_{NP}$
- Help: Can you help [me]$_{NP}$[with a flight]$_{PP}$
- Prefer: I prefer [to leave earlier]$_{TO\text{-}VP}$
- Told: I was told [United has a flight]$_{S}$
- …

Programming Analogy

- It may help to view things this way
  - Verbs are functions or methods
  - The arguments they take (subcat frames) they participate in specify the number, position and type of the arguments they take...
    - That is, just like the formal parameters to a method.

Summary

- CFGs appear to be just about what we need to account for a lot of basic syntactic structure in English.
- But there are problems
  - That can be dealt with adequately, although not elegantly, by staying within the CFG framework.
- There are simpler, more elegant, solutions that take us out of the CFG framework (beyond its formal power)
  - LFG, HPSG, Construction grammar, XTAG, etc.
  - Chapter 15 explores one approach (feature unification) in more detail

Treebanks

- Treebanks are corpora in which each sentence has been paired with a parse tree (presumably the right one).
- These are generally created
  1. By first parsing the collection with an automatic parser
  2. And then having human annotators hand correct each parse as necessary.
- This generally requires detailed annotation guidelines that provide a POS tagset, a grammar, and instructions for how to deal with particular grammatical constructions.

Penn Treebank

- Penn TreeBank is a widely used treebank.

Most well known part is the Wall Street Journal section of the Penn TreeBank.

- 1 M words from the 1987-1989 Wall Street Journal.

```
( (S (`` ``)
    (S-TPC-2
      (NP-SBJ-1 (PRP We) )
      (VP (MD would)
        (VP (VB have)
          (S
            (NP-SBJ (-NONE- *-1) )
            (VP (TO to)
              (VP (VB wait)
                (SBAR-TMP (IN until)
                  (S
                    (NP-SBJ (PRP we) )
                    (VP (VBP have)
                      (VP (VBN collected)
                        (PP-CLR (IN on)
                          (NP (DT those)(NNS assets))))))))))))
    (, ,) (`` ``)
    (NP-SBJ (PRP he) )
    (VP (VBD said)
      (S (-NONE- *T*-2) ))
    (. .) ))
```


Treebank Grammars

- Treebanks implicitly define a grammar for the language covered in the treebank.
- Simply take the local rules that make up the sub-trees in all the trees in the collection and you have a grammar
  - The WSJ section gives us about 12k rules if you do this
- Not complete, but if you have decent size corpus, you will have a grammar with decent coverage.

Treebank Grammars

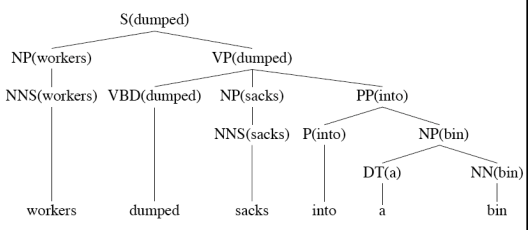
- Such grammars tend to be very flat due to the fact that they tend to avoid recursion.
  - To ease the annotators burden, among things
- For example, the Penn Treebank has ~4500 different rules for VPs. Among them...

```
VP → VBD PP
VP → VBD PP PP
VP → VBD PP PP PP
VP → VBD PP PP PP PP
```


Head Finding

- Finding heads in treebank trees is a task that arises frequently in many applications.
  - As we'll see it is particularly important in statistical parsing
- We can visualize this task by annotating the nodes of a parse tree with the heads of each corresponding node.

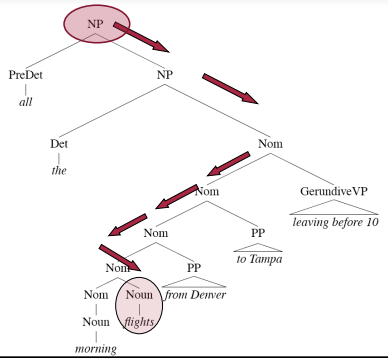
Lexically Decorated Tree



Head Finding

- Given a tree, the standard way to do head finding is to use a simple set of tree traversal rules specific to each non-terminal in the grammar.

Noun Phrases



Treebank Uses

- Treebanks (and head-finding) are particularly critical to the development of statistical parsers
  - Chapter 14
- Also valuable to *Corpus Linguistics*
  - Investigating the empirical details of various constructions in a given language

Parsing

- Parsing with CFGs refers to the task of assigning proper trees to input strings
- Proper here means a tree that covers all and only the elements of the input and has an S at the top
- It doesn't mean that the system can select the correct tree from among all the possible trees

Automatic Syntactic Parse