1. Syntax

By grammar, or 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 explicit instruction.

Not the kind of stuff you were later taught in “grammar” school:

- At least not in English-speaking countries :-)
- Indeed some EFL teaching involves something much closer to what we have in mind here.

2. Syntax (or Grammar)

Refers to the way words can be arranged in a given language.

Grammars (and parsing) are key components in many applications:

- Grammar checkers
- Dialogue management
- Question answering
- Information extraction
- Machine translation

3. Syntax, cont'd

There's a useful (traditional) contrast between two perspectives on this topic:

**paradigmatic**
What's interchangeable with what?
- words, phrases, . . .

**syntagmatic**
What co-occurs with what?
- ordering (before/after)
- marking (prefixes/suffixes)
Key notions that we’ll cover

- Categories (paradigmatic)
- Constituency (syntagmatic)
- Heads (syntagmatic)

Key formalism

- Context-free grammars

4. Constituency

Groups of words can be shown to act as single units, called **constituents**

In a given language, these units form coherent classes that behave in similar ways, w.r.t.

**External behavior**

How they relate to other units in the language

- We can say that in English, noun phrases can come before verbs

**Internal structure**

We can describe an internal structure to the class

- This might involve disjunctions of somewhat unlike sub-classes to do this
- For example, noun phrases can consist of a pronoun, a proper noun, or a complex phrase including a common noun

5. Constituency, cont’d: Noun Phrases

We can observe some commonality over the behaviour of the following English phrases:

<table>
<thead>
<tr>
<th>they</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cambodia</td>
</tr>
<tr>
<td>my aunt's pen</td>
</tr>
<tr>
<td>the reason I can't stay</td>
</tr>
<tr>
<td>taking another look at <em>Moby Dick</em></td>
</tr>
<tr>
<td>three french hens</td>
</tr>
</tbody>
</table>

One piece of evidence is that they can all precede verbs

- That is, occur in a frame such as

  ___________ surprised him.

- This is external, paradigmatic evidence

6. Grammars and Constituency

There’s nothing easy or obvious about how we come up with

- the ‘right’ set of constituents
- the rules that govern how they combine

That’s one of reasons there are so many different theories of grammar and competing analyses of the same data
The approach we'll explore isn't exactly "cutting-edge"

- But it's a good compromise between simplicity and adequacy
- And the technology required to support it is a good introduction to what's needed for most other approaches

7. Context-Free Grammars (CFGs)

Also known as **phrase structure grammars**

- And **Backus-Naur form** is a standardised approach to notating CFGs

For our purposes, a CFG consists of

**Terminals**

or **terminal symbols**: words (for now)

**Non-terminals**

or **non-terminal symbols**: Names for constituents in a language

- E.g., **NP** (noun phrase), **VP** (verb phrase), **V** (verb), **S** (sentence)

**Rules**

or **productions**, each of which is a pair of

- **a left-hand side**: a single non-terminal
- **a right-hand side**: a sequence of any number of terminals and non-terminals

For example: **VP → V NP**

**Distinguished symbol**

One of the non-terminals

- The starting point for all analyses
- Usually **S**

8. Some preliminary NP Rules

Some overly-simple rules for noun phrases:

- **NP → Det Nominal**
- **NP → ProperNoun**
- **Nominal → Noun | Nominal Noun**

These rules 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:

- A disjunction
  - Two kinds of nominals
  - Not strictly speaking a rule
  - Rather a shorthand notation for two rules
• A recursive definition
  ◦ Same non-terminal on the right and left-side of the rule

9. Defining a language with a grammar

What is the language defined by a grammar?

For FSAs, a string is in the language defined by a finite-state automaton F iff there is a non-empty sequence S of states of F such that

- The first state in S is a start state
- The last state in S is a terminal state
- Either
  ◦ The sequence is of length one and the string is empty
  ◦ There is at least one transition between every adjacent pair of states in S with a label such that the concatenation of all those labels is the string

10. Define a language with a CFG

There are two (main) ways to interpret a CFG as defining a language

A string is in the language defined by a context-free grammar G iff

Rewriting
You can get to the string by

- writing down G's distinguished symbol
- repeatedly writing down a new line by choosing a non-terminal from the line above, choosing a rule from G with that symbol as its left-hand side, then re-writing the line above with the chosen symbol replaced by the right-hand side of the chosen rule
- The resulting sequence is called a derivation

11. CFG interpretation, cont'd

Node admissability
A string is in a language defined by a grammar G iff there is at least one labelled tree such that

- The leaf nodes of the tree, in order, correspond to the string
- The root of the tree is labelled with G's distinguished symbol
- For every non-leaf node, there is a rule in G
  ◦ whose left-hand side is the label of the node
  ◦ whose right-hand side corresponds to the labels of the node's children, in order
12. Trees as proofs

Under either interpretation, we can use a parse tree (strictly speaking an ordered tree) to illustrate the way in which a string belongs to the language defined by a CFG.

For example, using the grammar and lexicon for the ATIS domain given in section 12.2 of Jurafsky & Martin (2nd edition)

- For the sentence
  - I prefer a morning flight

The parse tree that 'proves' that this is in the language is

![Parse Tree Image]

13. Generativity

As with FSAs and FSTs (see Lectures 2,3), you can view these rules as either analysis or synthesis machines

- Generate strings in the language
- Reject strings not in the language (recognition)
- Show how strings are in the language (parsing)

14. Parsing

Parsing is the process of taking a string and a grammar and returning one or more parse trees for that string.
Analogous to running a finite-state transducer over a tape

- But since CFGs are more powerful
  - That is, there are languages we can capture with CFGs that we can’t capture with finite-state methods
  - The parsing process is likewise more complicated
    - As we’ll see next week

15. A bit more detail on English Grammar

Huddleston and Pullum's *The Cambridge Grammar of the English Language* is 1860 pages long

- 

So we won’t cover all of English by a very long way

- Just enough to uncover some key shortcomings of CFGs

We’ll look briefly at

- Sentences
- Noun phrases
  - Agreement
- Verb phrases
  - Subcategorisation

16. Sentence Types

Declaratives

- A plane left
  

Imperatives

- Leave!
  

Yes-no questions

- Did the plane leave?
  

WH questions

- When did the plane leave?
  

17. Noun Phrases, more carefully

We can identify three quite distinct types of noun phrases:
Pronyms

she, he, we, ...

Proper Nouns

Edinburgh, Star Wars, the Eiffel Tower, ...

Complex noun phrases

the next prime minister after Thatcher

Consider the following moderately complicated noun phrase:

the first three morning flights from Denver to Tampa leaving before 10

We'll need something along the lines of the following tree:

18. NP Structure

That big NP is really about flights

• That’s its central critical noun
• Let’s call that the head of the NP

We can dissect this kind of NP into:

• The constituents that can come before the head
• The constituents that can come after it

19. Before the nominal: Determiners

Complex noun phrases can start with determiners
Determiners can be

**Simple lexical items**

- the, this, a, her

**(Recursive) possessives**

- **simple** Robin’s car
- **complex** Robin’s youngest child’s toy

### 20. Before the nominal: Other premodifiers

Other premodifiers include

- Quantifiers, cardinals, ordinals:
  - **every flight**
  - **three flights**
  - **first flight**

- Adjectives and Adjective phrases:
  - **large cars**
  - **extremely sleepy baby**

There are constraints we haven’t captured on the *order* of pre-modifiers:

- Between adjectives and quantifiers:
  - *eligible every candidate*
  - **every eligible candidate**

- Between one adjective and another:
  - **big red bus**
  - **red big bus**

- Following a common linguistic convention, I’m using an initial asterisk to indicate a word sequence which is *not* in a (natural) language or cannot (should not) be accepted by a formal grammar
- Likewise an initial question mark for a borderline in/out word sequence

### 21. The nominal: the head and its postmodifiers

Eventually (or even right away), we get to the Nominal
• Including the head, with or without compounding

<table>
<thead>
<tr>
<th>CNP</th>
<th>Nominal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal</td>
<td>Noun</td>
</tr>
<tr>
<td>Nominal</td>
<td>Nominal Noun</td>
</tr>
</tbody>
</table>

The postmodifiers which stack up behind the head may include

• Prepositional phrases: flight from Seattle

• Non-finite clauses (gerundive, infinitive): flights arriving before noon first flight to depart

• Relative clauses: flights that serve breakfast people whom the pilot knows

Similar general (recursive) rules to handle these

<table>
<thead>
<tr>
<th>Nominal</th>
<th>Nominal PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal</td>
<td>Nominal GerundVP</td>
</tr>
<tr>
<td>Nominal</td>
<td>Nominal InfVP</td>
</tr>
<tr>
<td>Nominal</td>
<td>Nominal RelClause</td>
</tr>
</tbody>
</table>