1 Motivation

What is syntax?

- The study of grammatical relations between words and other units within the sentence. For example: word order, agreement, predicate-argument relations.

- It goes down to the word level (below which is morphology) and up to the sentence level (above which is discourse analysis).

- It is concerned with structure, not meaning (which is what semantics deals with), but syntactic structure is the scaffolding for semantics

\[(1) \quad \text{The man who saw John rode the bicycle.}\]
\[(2) \quad man'(m) \land saw'(m, John') \land bicycle'(b) \land ride'(m, b)\]

What is formal syntax?

- Its goal is to explain how, for every (possible) human language, to come up with a mechanical procedure (i.e., a computer program) to (1) generate all possible sentences of that language, and (2) assign, to each sentence that does, a structural description

- Chomsky defined the terms weak generative capacity and strong generative capacity to talk about (1) and (2), respectively

What is computational syntax?

- Like formal syntax, but tightly grounded in theory of computation

- Also concerned with the real possibility of implementation in a computer

- Concerned with computational complexity: how fast would a computer program run that can accept strings (and assign structural descriptions to them) or reject them?

- More interested in language-in-use than extreme cases

\[(3) \quad \text{a. Who did John say that Mary wondered whether Bill liked?}\]
\[\text{b. *Who did you wonder why she wrote to?}\]

\[(4) \quad \text{Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.}\]
2 Overview of upcoming topics

- What kind of formal system do we need for natural language?
- Parsing algorithms for context-free grammars
- Using grammars to compute values, in particular, probabilities for probabilistic parsing
- But first...

3 Basic notation and definitions

- Alphabet: a finite set of symbols
- String: a possibly empty sequence of symbols drawn from an alphabet
  - Customarily, we use variables $w, v, u$ to range over strings.
  - The empty string is denoted $\epsilon$. It is not a symbol of an alphabet, but a metasymbol
  - We write $v \cdot w$ or just $vw$ for concatenation, and $(w^n)$ for repeated concatenation
  - True or false: $a \cdot b = b \cdot a$? $a(bc) = (ab)c$? $a \cdot \epsilon = a$? $a \cdot \epsilon = \epsilon \cdot a$? If $v = \epsilon$ and $w = ab$, $vw = w$?
  - What is $\epsilon^5$? $a^0$?

- Language: a possibly infinite, possibly empty set of strings over an alphabet
  - The empty language is $\emptyset$
  - True or false: $\emptyset = \{\epsilon\}$? $\{a\} \cup \{\epsilon\} = \{a\}$?

- Grammar: a finite description of a possibly infinite language

4 Finite-state machines

- Pro: very efficient (linear time) processing

- Example:

  (5) the white male hired another white male
  (6) the white male hired another white male who hired another white male
  (7) the white male hired another white male (who hired another white male)$^n$

- How about this:

  (8) the white male hired another white male
  (9) the white male whom a white male hired hired another white male
  (10) the white male (whom a white male)$^n$ hired$^n$ hired another white male

- It can be shown that $\{a^n b^n | n \geq 0\}$ and similar languages cannot be described by a finite-state machine
5 Context-free grammars

Overview

• Key idea: treat a group of words as a single unit (called a phrase or constituent)
• A CFG describes how larger constituents are formed out of smaller ones, or symbols
• We can parse (given an input string, decide whether it belongs to the language, and if so, assign a tree) in $O(n^3)$ time

Definition

• An alphabet of terminal symbols
• An alphabet of nonterminal symbols ($A, B, C, \ldots$ or $X, Y, Z, \ldots$)
• A special nonterminal symbol called the start symbol
• A set of productions

Example grammar:

\begin{align*}
(11) \quad S & \rightarrow aSb \\
(12) \quad S & \rightarrow \epsilon \\
\end{align*}

Example derivation:

\begin{align*}
(13) \quad S & \Rightarrow aSb & \text{using rule (11)} \\
(14) \quad & \Rightarrow aaSbb & \text{using rule (11)} \\
(15) \quad & \Rightarrow aaaSbbb & \text{using rule (11)} \\
(16) \quad & \Rightarrow aaabbb & \text{using rule (12)} \\
\end{align*}

• As a tree:

\begin{align*}
(17) \quad \text{S} \quad \text{S} \quad \text{S} \\
\quad \text{a} \quad \text{a} \quad \text{a} \\
\quad \text{S} \quad \text{S} \quad \text{S} \\
\quad \text{b} \quad \text{b} \quad \text{b} \\
\quad \text{S} \quad \text{S} \quad \text{S} \\
\quad \text{b} \quad \text{b} \\
\quad \text{S} \quad \text{S} \\
\quad \epsilon \\
\end{align*}

• In general, this grammar generates $\{a^n b^n \mid n \geq 0\}$
6 A CFG for a little bit of English

Preliminaries

• Terminal alphabet: words
• Nonterminal alphabet: syntactic categories
• Strings: sentences
• Language: English

Basic clause structure

(18) \( S \rightarrow \text{NP VP} \)
(19) \( \text{VP} \rightarrow \text{V NP} \)
(20) \( \text{NP} \rightarrow \text{the dog} \)
(21) \( \text{NP} \rightarrow \text{the man} \)
(22) \( \text{V} \rightarrow \text{bit} \)

Subcategorization

• Some more rules:

(23) \( \text{V} \rightarrow \text{existed} \)
(24) \( \text{V} \rightarrow \text{put} \)

• But what about:

(25) *The man existed the dog.
(26) *The man put the dog.

• The fix:

(27) \( \text{VP} \rightarrow V[subcat = \langle \text{NP} \rangle] \text{NP} \)
(28) \( \text{VP} \rightarrow V[subcat = \langle \rangle] \)
(29) \( \text{VP} \rightarrow V[subcat = \langle \text{NP}, \text{PP} \rangle] \text{NP PP} \)
(30) \( V[subcat = \langle \text{NP} \rangle] \rightarrow \text{bit} \)
(31) \( V[subcat = \langle \rangle] \rightarrow \text{existed} \)
(32) \( V[subcat = \langle \text{NP}, \text{PP} \rangle] \rightarrow \text{put} \)
(33) \( \text{PP} \rightarrow \text{P NP} \)
(34) \( \text{P} \rightarrow \text{in} \)
(35) \( \text{NP} \rightarrow \text{house} \)
Agreement

• More rules:

(36)  \( N \rightarrow \text{men} \)
(37)  \( N \rightarrow \text{dogs} \)
(38)  \( N \rightarrow \text{houses} \)
(39)  \( V \rightarrow \text{bite} \)
(40)  \( V \rightarrow \text{bites} \)

• But what about:

(41)  *The dogs bites the man.
(42)  *The dog bite the man.

• Fix:

(43)  \( S \rightarrow \text{NP[\text{num} = x]} \text{ VP[\text{num} = x]} \quad x \in \{s, p\} \)
(44)  \( \text{VP[\text{num} = x]} \rightarrow \text{V[\text{num} = x]}\text{NP[\text{num} = y]} \)
(45)  \( \text{NP[\text{num} = s]} \rightarrow \text{dog} \)
(46)  \( \text{NP[\text{num} = p]} \rightarrow \text{dogs} \)
(47)  \( \text{V[\text{num} = s]} \rightarrow \text{bites} \)
(48)  \( \text{V[\text{num} = p]} \rightarrow \text{bite} \)

Expletive ‘there’

• How about:

(49)  There exists a dog.
(50)  *There exist a dog.
(51)  *The dog exists the man.
(52)  *There exists.

• Analysis:

(53)  \( S \rightarrow \text{NP[\text{there}]} \text{ VP[\text{there}]} \)
(54)  \( \text{NP[\text{there}]} \rightarrow \text{there} \)
(55)  \( \text{VP[\text{there}]} \rightarrow \text{V[\text{subcat} = \langle \rangle, \text{num} = x]} \text{NP[\text{num} = x]} \)
(56)  \( \text{V[\text{subcat} = \langle \rangle, \text{num} = s]} \rightarrow \text{exists} \)
(57)  \( \text{V[\text{subcat} = \langle \rangle, \text{num} = p]} \rightarrow \text{exist} \)