Parsing: Phrase-Structure Grammars to Dependency Grammars

David Israel
SRI (Emeritus)
Sapienza (Visiting)
Remember last time ...

- $V \times T$
- Where $V$ is a complicated Vector Space
  - A Tensor Algebra
  - Which is where the semantics lives
- And $T$?
- $T$ is a syntactic algebra of a certain sort, which one can think of as a way of representing a phrase-structure grammar
  - Kind of ....
In the Beginning, there was Chomsky

- Phrase-structure (rewrite) rules:
- Stated in terms of typed constituents
  - \( S \to NP \ VP \)
  - \( NP \to (\text{Det}) \ N \)
  - \( N \to (\text{AP}) \ N \ (\text{PP}) \)
  - \( VP \to V_{\text{Trans}} \ NP \)

- And a lexicon
  - \( N = \{\text{David, Roma, man} \ldots\} \)
  - \( \text{Adj} = \{\text{bald, beautiful} \ldots\} \)
  - \( V_{\text{Trans}} = \{\text{loves,} \ldots\} \)
  - \( \text{Det} = \{\text{a, the,} \ldots\} \)
  - \( P = \{\text{in, from, of} \ldots\} \)
Phrase Structure/Constituent Trees

The bald man loves beautiful Roma
The Chomsky Hierarchy: 
The General Set-up

• Finite set of production rules, whose left- and right-hand sides consist of sequences of symbols from

• Finite set of nonterminal symbols, including a distinguished element $S$ (Sentence – the Start symbol)

• Finite set of terminal symbols (the lexicon/vocabulary) – which might include a distinguished element $\varepsilon$, for the empty string
The Hierarchy:  
From Most to Least Restrictive

- **Type-3 (finite-state) grammars:**
  - Single nonterminal on left
  - Single terminal on right, possibly *followed* by a single nonterminal (right regular)
    - or --- exclusive “or”
  - Single terminal on right, possibly *preceded* by a single nonterminal (left regular)
    - \( S \rightarrow \varepsilon \), if \( S \) does not occur on right of any rule

- Generate *regular languages*
- Recognized/Decided by finite-state automata
Type-2 Grammars: Context-Free Grammars

- Single nonterminal on left
- String of terminal and/or nonterminals on right
- Generate context-free languages
- Recognized/decided by non-deterministic push-down automata
Type-1 Grammars: Context-Sensitive Grammars

- $\alpha A \beta \rightarrow \alpha \gamma \beta$: $A$ a nonterminal, $\alpha$, $\beta$, $\gamma$ strings of nonterminals and/or terminals
- $\alpha$ and $\beta$ may be empty, but not $\gamma$
- $S \rightarrow \varepsilon$, if $S$ does not occur on right of any rule
- Generates the set of context-sensitive languages
- Recognized/decided by linear bounded automaton: nondeterministic Turing machine is tape-length is bounded by some constant times length of input
Type-0 Grammars: Unrestricted

- No restrictions (What a surprise!)
- Generate all languages recognizable by a Turing machine, so
- All recursively enumerable languages
- Includes languages where membership is undecidable – r.e., but not recursive
- Can’t always tell in a finite computation when some string that isn’t in the language isn’t in the language
And English (Italian, etc., etc.,...)?

• What type? For an empirically adequate grammar
  – Mildly context-sensitive
  – Huh?
  – Just beyond context-free, but nowhere near the full power of context-sensitive grammars
  – Hard to define precisely (or easily) in terms of the form of the grammars in the form of standard production rules

• How many nonterminals needed?
  – Who knows?
Phrase Structure Parsing

- Structural decomposition of a sentence – given as a sequence of words
- Into constituents or *brackets*
- Generally applicable data structure: nested trees
- Enormous Amount of Ambiguity
  - Even small grammars (10 rules) + tiny lexicons generate 100’s of parses per sentence
Syntactic Ambiguities

- impractical design requirements
- plastic cup holder
- David cleaned the dishes from dinner
- David cleaned the dishes with detergent
- David cleaned the dishes in his tuxedo
- David cleaned the dishes in the sink
- David cooked the beans in the pot in the stove with handles
- Visiting relatives can be boring
- Changing schedules frequently confused customers
- The chicken is ready to eat
- The lawyer is rich enough to sue
- Small rats and mice can squeeze through cracks in the walls
- Etc., etc., etc.
Hand-written Broad Coverage CFGs

- ARE IMPOSSIBLE!!
- Well, ... They’re very, very big
- Very hard to add to, edit, maintain
- Can generate millions of parses for average-length sentences
- And still don’t have broad enough coverage
- As a species, they are *almost* extinct
(Probabilistic) CFGs

- A CFG as a 4-tuple \(<N, T, S, R>\)
  - \(N\): set of nonterminals
    - Phrasal categories: S, NP, VP, AdjP, etc.
    - POS (pre-terminals): N, V, Adj, etc.
  - \(T\): set of terminals (lexicon)
  - \(S\): distinguished nonterminal start symbol
    - Sometimes another special nonterminal is used, e.g., ROOT
  - \(R\): the set of (cfg) rules or productions
- A PCFG adds: A top-down production probability per rule
  - Where: \(X \rightarrow Y_1 Y_2 \ldots Y_n \in R, \text{Prob}(Y_1 Y_2 \ldots Y_n | X)\)
Unsupervised Learning of PCFG’s

• Significant efforts begin in the early ‘90’s
  – Carroll and Charniak (‘92); Pereira and Schabes (‘92); Brill (‘93); Stolcke and Omohundro (‘94)

• Penn Treebank (early 90’s) as a crucial resource
  – Large corpus (bank?) of trees: of phrase-structure analyses of English sentences
  – Very carefully trained annotators
  – Demand for high inter-annotator agreement (>> 90%)
Treebank Sentences

(S (NP-SBJ The move)
 (VP followed
  (NP (NP a round)
   (PP of
    (NP (NP similar increases)
     (PP by
      (NP other lenders))
     (PP against
      (NP Arizona real estate loans))))))
,
(S-ADV (NP-SBJ *)
 (VP reflecting
  (NP (NP a continuing decline)
   (PP-LOC in
    (NP that market))))))
A Simple(!) PCFG

(a)

S → NP VP
VP → V NP
VP → V NP PP
NP → Det Noun
NP → NP PP
PP → P NP
V → saw
V → prodded
N → telescope
N → stick
N → girl
N → boy
(1)
(.25)
(.7)
(.3)
(.3)
(.1)
(.8)
(.2)
(.2)
(.3)
(1)
(1)
(1)
(.1)
A Parse of
“The girl saw the boy with the telescope.”
Yet Another Parse

Pr(tree) = 1 x .7 x 1 x .3 x .75 x .8 x .3 x .7 x 1 x .1 x 1 x 1 x .7 x 1 x .2 ≈ 0.00037
Unsupervised Learning of Dependency Grammar

• Production Rules for DGs:
  – Same set of nonterminals and terminals as PSG’s
    • SO, depends in a minor way on which PSG you’re dealing with
    • Note: **ROOT** usually replaces **S** as distinguished start symbol
  – \( X \rightarrow XY \) or \( X \rightarrow YX \)
  – And that’s it: all dependencies are binary
  – Often in learning there are no lexical items; the terminals are, e.g., POS tags or word-class labels
  – No big dependency-annotated corpus for English ... yet.