Dealing with Ambiguity

- Consider possible parses but weighted by probability
- Return likeliest parse
- Return likeliest parse along with a probability

PCFG: Probabilistic Context-Free Grammar

- Components of PCFG: $G = \langle N, \Sigma, R, S \rangle$
  - Σ, an alphabet or set of *terminal* symbols
  - N, a set of *nonterminal* symbols, $N \cap \Sigma = \emptyset$
  - $S \in N$, a *start* symbol (distinguished nonterminal)
  - R, a set of *rules* or *productions* of the form

$$A \longrightarrow \beta\,[p]$$

    - $A \in N$ is a single nonterminal and $\beta \in (\Sigma \cup N)^*$ is a finite string of terminals and nonterminals
    - $p = P(A \longrightarrow \beta | A)$ is the probability of expanding A to β

$$\sum_{\beta} P(A \longrightarrow \beta | A) = 1$$

- Consistency:
  - Probability of a sentence is nonzero if and only if it is in the language
  - Sum of probabilities of sentences in the language is 1

Languages from Grammars

- Simple CFG: Nominal is the start symbol

$$\begin{aligned} \text{Nominal} &\longrightarrow \text{Nominal Noun} \\ \text{Nominal} &\longrightarrow \text{Noun} \\ \text{Noun} &\longrightarrow \text{olive} \\ \text{Noun} &\longrightarrow \text{jar} \end{aligned}$$

- Simpler CFG: Nominal is the start symbol

$$\begin{aligned} \text{Nominal} &\longrightarrow \text{Nominal Noun} \\ \text{Noun} &\longrightarrow \text{olive} \\ \text{Noun} &\longrightarrow \text{jar} \end{aligned}$$

- Simple PCFG: Nominal is the start symbol

$$\begin{aligned} \text{Nominal} &\longrightarrow \text{Nominal Noun } [\tfrac{2}{3}] \\ \text{Nominal} &\longrightarrow \text{Noun } [\tfrac{1}{3}] \\ \text{Noun} &\longrightarrow \text{jar } [1] \end{aligned}$$

Consistent PCFG

Probability of the language is 1

- Consider the same simple PCFG as before

$$\begin{array}{rcl} \text{Nominal} & \longrightarrow & \text{Nominal Noun } [\frac{2}{3}] \\ \text{Nominal} & \longrightarrow & \text{Noun } [\frac{1}{3}] \\ \text{Noun} & \longrightarrow & \text{jar } [1] \end{array}$$

- Write out all parse trees for jar^k
- Probability of jar^k is sum of probabilities for its parse trees
- Sum up the probabilities for the entire language

Inconsistent PCFG

Probability of generating the language is not 1

- Consider a modified PCFG: Nominal is the start symbol

$$\begin{aligned} \text{Nominal} &\longrightarrow \text{Nominal Nominal } [\tfrac{2}{3}] \\ \text{Nominal} &\longrightarrow \text{jar } [\tfrac{1}{3}] \end{aligned}$$

- Write out all parse trees for jar^k
- Probability of jar^k is sum of probabilities for its parse trees
- Sum up the probabilities for the entire language

The argument gets cumbersome

PCFG: Markovian Argument

- Consider how a derivation proceeds
  - One production increases the count of nonterminals by one
  - One production decreases the count of nonterminals by one
  - We start with one nonterminal (the start symbol)
  - Any derivation that ends in zero nonterminals yields a string in the language
- $L(n+1)$ (left move): probability of starting from $n+1$ nonterminals and arriving at a state with n nonterminals
  The probability of generating a string in this language is $L(1)$
- $L(0)$ is never used and could be left undefined or set to zero
- PCFGs respect the Markov assumption: any nonterminal has an equal chance of being expanded regardless of history
- Therefore, $L(n+1)$ is a constant, L

Inconsistent PCFG: Markovian Derivation

- Probabilities of stepping right q and left $1-q$
- L (probability of eventually moving one left) equals
  - Stepping one left immediately plus
  - Stepping one right followed by two paths moving one step left each

$$L = 1 - q + qL^2$$

- Solve $qL^2 - L + 1 - q = 0$
- $L = \frac{1 \pm \sqrt{1-4q(1-q)}}{2q}$
- $\sqrt{1-4q(1-q)} = (2q-1)$
- Therefore, L has two solutions, of which the *minimum* is appropriate
  - Trivial solution: $L = \frac{1-(1-2q)}{2q} = 1$
  - Left-right odds: $L = \frac{1-(2q-1)}{2q} = \frac{1-q}{q}$
- For our example, $L = \min(1, \frac{\frac{1}{3}}{\frac{2}{3}}) = \frac{1}{2} \neq 1$—indicating inconsistency
- If we reverse the probabilities, then $\min(1,2) = 1$

Probability of a Parse Tree

- Tree T obtained from sentence W, i.e., T *yields* W

$$P(T,W) = P(T)P(W|T)$$

$$P(T,W) = P(T) \text{ since } P(W|T) = 1$$

- Obtaining T via n expansions $A_i \longrightarrow \beta_i$ and $S = A_1$ is the start symbol

$$P(T,W) = \prod_{i=1}^{n} P(\beta_i|A_i)$$

- Best tree for W

$$\widehat{T}(W) = \underset{T \text{ yields } W}{\operatorname{argmax}} P(T|W) = \underset{T \text{ yields } W}{\operatorname{argmax}} \frac{P(T,W)}{P(W)}$$

- Since $P(T,W) = P(T)$ and $P(W)$ is constant (W being fixed)

$$\widehat{T}(W) = \underset{T \text{ yields } W}{\operatorname{argmax}} P(T)$$

Probabilistic CKY Parsing

- Like CKY, as discussed earlier, except that
  - Each cell contains not a set of, but a probability distribution over, nonterminals
- Specifying probabilities for Chomsky Normal Form
  - Consider each transformation used in the normalization
- Supply the probabilities below
  - Replace $A \longrightarrow \alpha B \gamma[p]$ and $B \longrightarrow \beta[q]$ by $A \longrightarrow \alpha \beta \gamma[?]$
  - Replace $A \longrightarrow BC\gamma[p]$ by $A \longrightarrow BX[?]$ and $X \longrightarrow C\gamma[?]$
- Store a probability distribution over nonterminals in each cell
- Return likeliest parse

Learning PCFG Probabilities

- Simplest estimator: Assume a treebank
- Estimate the probability of $A \longrightarrow \beta$ as

$$P(A \longrightarrow \beta | A) = \frac{\text{Count}(A \longrightarrow \beta)}{\sum_{\gamma} \text{Count}(A \longrightarrow \gamma)} = \frac{\text{Count}(A \longrightarrow \beta)}{\text{Count}(A)}$$

- Without a treebank but with a corpus
- Assume a traditional parser
- Initialize all rule probabilities as equal
- Iteratively
  - Parse each sentence in the corpus
  - Credit each rule $A \longrightarrow \beta_i$ by the counts weighted by the probabilities of the rules leading to that nonterminal, A
  - Revise the probability estimates
- More properly described as an expectation maximization algorithm

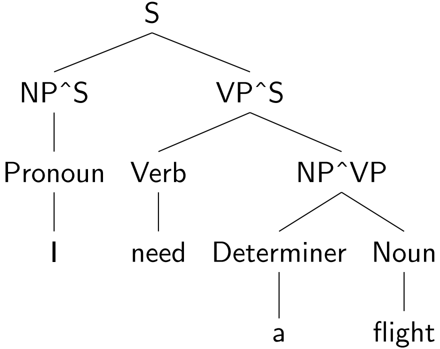
Shortcomings of PCFGs

PCFGs break ties between rules in a fixed manner

- Naïve context-free assumption regarding probabilities
  - NP $\stackrel{*}{\longrightarrow}$ Pronoun much likelier for a Subject NP than an object NP
  - PCFGs (and CFGs) disregard the path on which the NP was produced
- Lack of lexical dependence
  - VP $\longrightarrow$ VBD NP NP is likelier for a ditransitive verb
- Consider prepositional phrase attachment
  - Either: prefer PP attached to VP ("dumped sacks into a bin")
    - VP $\longrightarrow$ VBD NP PP
  - Or: prefer PP attached to NP ("caught tons of herring")
    - VP $\longrightarrow$ VBD NP
    - NP $\longrightarrow$ NP PP
- Coordination ambiguities: each parse gets the same probability because all parses use the same rules

Split Nonterminals to Refine a PCFG

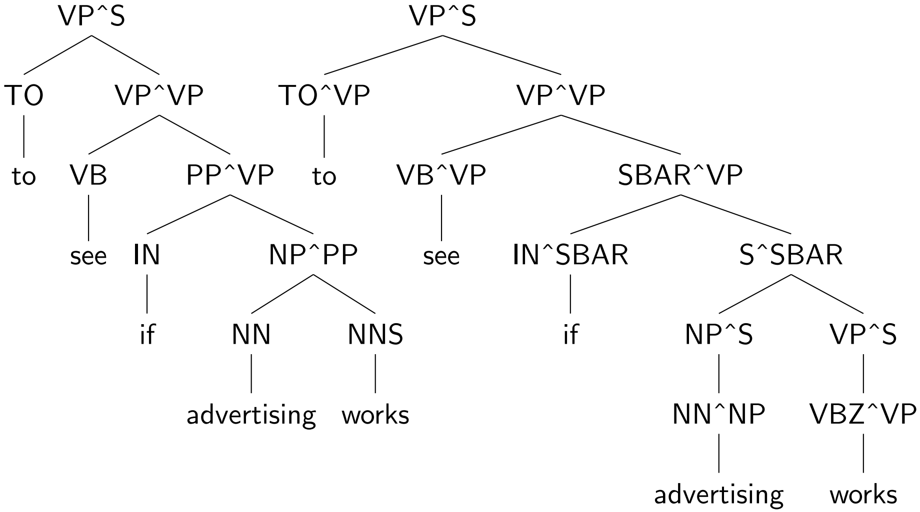
- Split nonterminals for syntactic roles, e.g., $NP_{subject}$ versus NP_{object}
  - Then learn different probabilities for their productions
- Capture part of path by a parent annotation
  - Annotating only the phrasal nonterminals (NP^S versus NP^VP)



  - Likewise, split *preterminals*, i.e., nonterminals that yield terminals
    - Adverbs depend on where they occur: RB^AdvP (also, now), RB^VP (not), RB^NP (only, just)

Example of Preterminals with Sentential Complements

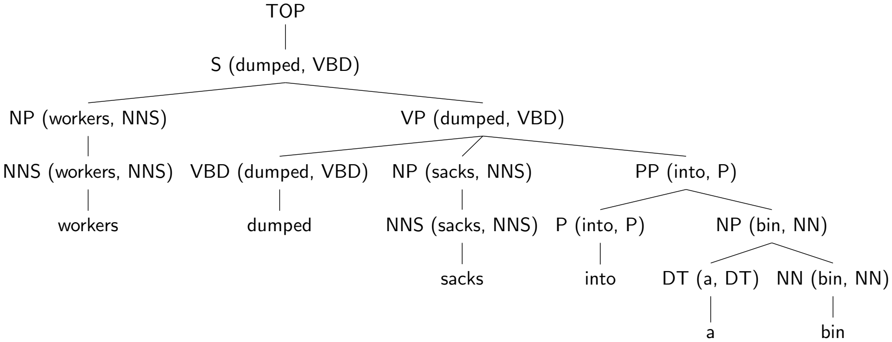
Klein and Manning: Left parse is wrong



IN includes preps, complementizers (that), subord conjs (if, as)

Lexicalized Parse Tree

Variant of previous such tree with parts of speech inserted



TOP	$\longrightarrow$	S(dumped, VBD)
S(dumped, VBD)	$\longrightarrow$	NP(workers, NNS) VP(dumped, VBD)
VP(dumped, VBD)	$\longrightarrow$	VBD(dumped, VBD) NP(sacks, NNS) PP(into, P)
...		
VBD(dumped, VBD)	$\longrightarrow$	dumped
...		

Estimating the Probabilities

- In general, we estimate the probability of $A \longrightarrow \beta$ as

$$P(A \longrightarrow \beta | A) = \frac{\text{Count}(A \longrightarrow \beta)}{\sum_\gamma \text{Count}(A \longrightarrow \gamma)} = \frac{\text{Count}(A \longrightarrow \beta)}{\text{Count}(A)}$$

- But the new productions are highly specific
- Collins Model 1 makes independence assumptions
  - Treat β as $\beta_1 \ldots \beta_H \ldots \beta_n$: β_H is the head and $\beta_1 = \beta_n = \textsc{stop}$
  - Generate the head
  - Generate its premodifiers until getting to STOP
  - Generate its post-modifiers until getting to STOP
  - Apply Naïve Bayes

$$P(A \longrightarrow \beta) = P(A \longrightarrow \beta_H) \times P(\beta_1 \ldots \beta_{H-1} | \beta_H) \times P(\beta_{H+1} \ldots \beta_n | \beta_H)$$

$$\approx P(A \longrightarrow \beta_H) \times \prod_{k=1}^{H-1} P(\beta_k | \beta_H) \times \prod_{k=H+1}^{n} P(\beta_k | \beta_H)$$

- Estimate each probability from smaller amounts of data

Labeled Recall and Precision to Evaluate Parsers

- Like recall and precision but
  - Based on counting correct constituents identified
  - Correctness with respect to a ground truth *reference* parse tree
- Recall
  - How many of the correct constituents are discovered
- Precision
  - How many of the constituents discovered are correct

Cross Brackets

A metric specific to comparing parse trees

- A measure of error
- The number of constituents for which
  - The reference parse has a bracketing ((A B) C)
  - The hypothesis parse has a bracketing (A (B C))
- On the Wall Street Journal treebank, modern parsers yield
  - Recall 90%
  - Precision 90%
  - Cross-bracketing 1%
- Extended metrics for comparing parsers using different grammars

Human Parsing

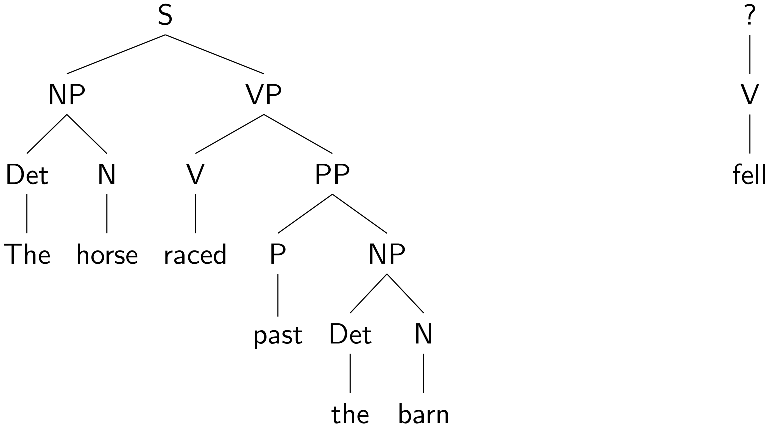
Psycholinguistics

- Studies of human processing ease
  - Delay in reading
  - Eye gaze fixation (dwell) time
- Garden-path sentences
  - Prefix (initial portion) is ambiguous
  - That is, temporarily ambiguous while reading
  - A higher preferred parse of the prefix doesn't lead to a parse of the entire sentence

The Horse Raced Past the Barn Fell: Problematic

A complete sentence followed by an extra verb

The first part gets a likely parse that offers no clear attachment for the final verb



The Horse Raced Past the Barn Fell: Correct

Raced is part of a reduced relative clause modifying "The horse"