ECE8813
Statistical Natural Language Processing

Lecture 23: Probabilistic Context Free Grammar

Chin-Hui Lee

School of Electrical and Computer Engineering

Georgia Institute of Technology

Atlanta, GA 30332, USA

chl@ece.gatech.edu

Phrase Structure

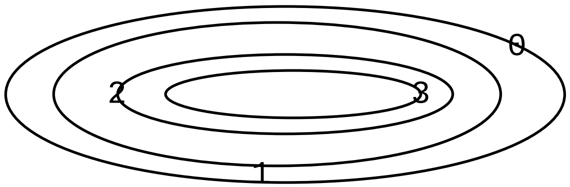
- Syntax and word order
  - "I want to go to a movie tomorrow." (English vs. Chinese)
- Constituents and phrases: equivalent classes
  - Noun phrases
  - Verb phrases
  - Prepositional phrases
  - Adjective phrases
- Phrase structure grammars
  - Start symbols and derivation (rewrite) rules
  - Terminal vs. non-terminal nodes
  - Local vs. global parse trees
  - Dependency: arguments and adjuncts
- Semantics (meaning) and pragmatics
- Language-specific properties: Multilingual issues

Chunking and Grammar Induction

- **Chunking**: recognizing higher level units of structure that allow us to compress our description of a sentence
- **Grammar Induction**: Explain the structure of chunks found over different sentences
- **Parsing**: can be considered as implementing chunking and discovering sentence structures

Formal Grammar Specification

- Grammar *G={A, I, S, D}* and Language *L(G)*
  - G is defined by an alphabet set *A*, an intermediate set *I*, a root symbol *S*, and a set of derivation (production) rules *D*
  - *L(G)* is the language of the set of sentences generated by *G*
- Type of String Grammars



  - *Type 0: free* or *unrestricted*
  - *Type 1: context-sensitive*

$$D=\{\alpha\theta\beta \rightarrow \alpha\psi\beta\} \quad \theta \in I \quad \psi \in I \cup A \quad \alpha,\beta : \text{string}$$

  - *Type 2: context-free*

$$D=\{\theta \rightarrow \psi\} \quad \theta \in I \quad \psi \in I \cup A$$

  - *Type 3: finite state* or *regular* $D=\{\alpha \rightarrow z\beta, \alpha \rightarrow z\} \quad \alpha,\beta \in I \quad z \in A$
- *Chomsky Normal Form* (CNF)
  - a context-free language can be replaced by another language in CNF

$$D=\{\alpha \rightarrow \beta\gamma, \alpha \rightarrow z\} \quad \alpha,\beta,\gamma \in I \quad z \in A$$

Context Normal Form (CNF)

- Chomsky hierarchy
  - Type 0 Grammars/Languages
    - rewrite rules $\alpha \rightarrow \beta$; α,β: any string of terminals and nonterminals
  - Context-sensitive Grammars/Languages
    - rewrite rules: $\alpha X \beta \rightarrow \alpha\gamma\beta$, where X is nonterminal, a,b,g any string of terminals and nonterminals (g must not be empty)
  - **Context-free Grammars/Languages**
    - rewrite rules: $X \rightarrow \gamma$, where X is nonterminal, γ any string of terminals and nonterminals, *G={A, I, S, D}* and Language *L(G)*
  - Regular Grammars/Languages
    - rewrite rules: $X \rightarrow \alpha\ Y$ X,Y: nonterminals, *a:* terminal string

Context-Free Grammars

- A context free grammar consists of a set of phrase structure rules:
  - Examples
    - S $\rightarrow$ NP VP
    - N $\rightarrow$ dog
  - One left hand side symbol (non-terminal)
  - A sequence of right hand side symbols (terminals or non-terminals)
  - "Context-Free" means that the LHS symbol of a rule can be rewritten as the sequence of RHS symbols in any context

Another NLP Example

#1 S → NP VP

#2 VP → V NP PP

#3 VP → V NP

#4 NP → N

#5 NP → N PP

#6 PP → PREP N

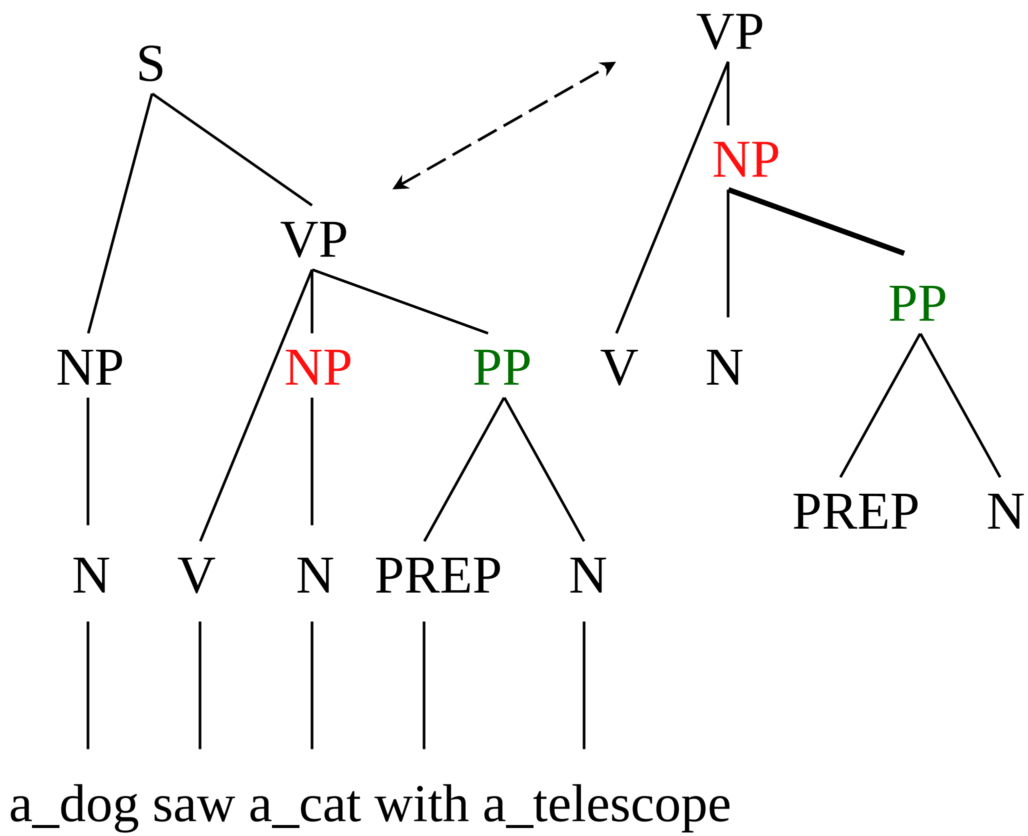
#7 N → a_dog

#8 N → a_cat

#9 N → a_telescope

#10 V → saw

#11 PREP → with

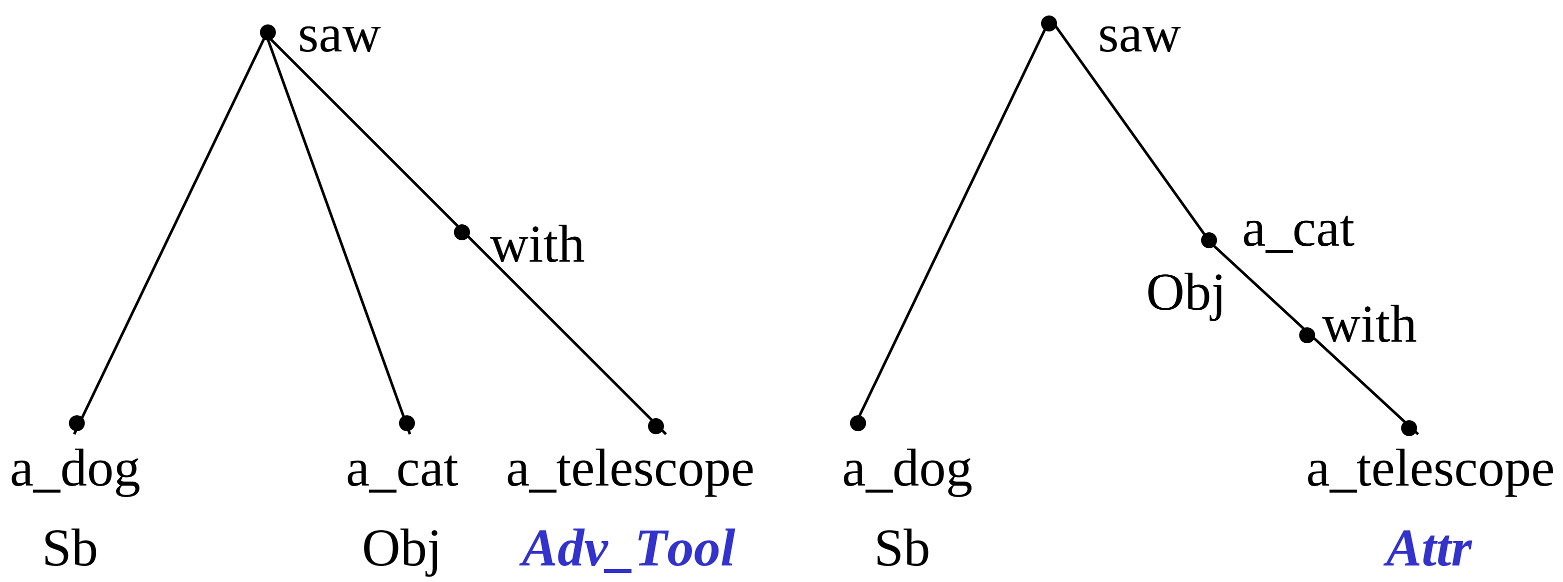


Phrases & Dependency Grammars

- In a dependency grammar, one word is the head of a sentence, and all other words are either a dependent of that word, or else dependent on some other word which connects to the head word through a series of dependencies
  - Lexicalized: Dependencies between words are taken care of to Include more information about the individual words when making decisions about the parse tree structure
  - A way of decomposing phrase structure rules

Dependency Style Example

- Same example, dependency representation



Assumptions

- Independence assumptions (very strong!)
- Independence of context (neighboring subtrees)
- Independence of ancestors (upper levels)
- Place-independence (regardless where in tree it appears) ~ time invariance in HMM

Probability of a Derivation Tree

- Both phrase/parse/derivational “grammatical”
- Different meaning: which is better [in context]?
- “Internal context”: relations among phrases, words
- Probabilistic CFG:

relations among a mother node & daughter nodes

in terms of expansion [rewrite,derivation] probability

define probability of a derivation (i.e. parse) tree:

$$P(T) = \Pi_{i=1..n}\, p(r(i))$$

r(i) are all rules of the CFG used to generate the sentence W of which T is a parse

Probabilistic Context Free Grammar

PCFG: G = {A, I, S, D, P(D)}

$$A = \{w_1, \ldots, w_V\}$$

$$I = \{I_1, \ldots, I_Q\} \quad \text{with} \quad S = I_1$$

$$D = \{I_i \rightarrow H_j\} \quad \text{with} \quad j = 1, \ldots, J_i$$

$$\forall i \quad \sum_{j=1}^{J_i} P(I_i \rightarrow H_j) = 1 \quad H_j = \text{symbol-sequence}$$

- Probability of a word sequence *W* according to *G*

$$P(W \mid G) = P(w_1^M \mid G) = \sum_t P(w_1^M \mid t) P(t) \quad t: \text{parse-tree}$$

- Probability of a parse tree (score and compare)

$$P(t) = P(d_1^L) = \prod_{j=1}^{L} P(d_j) \quad d_j: \text{parse-tree-rule}$$

Center of Signal and Image Processing
Georgia Institute of Technology
CSIP

Properties of PCFG

- Place Invariance
  - Probability of a subtree does not depend on where in the sentence it dominates (spanning from *p* to *q*)

$$P(I_j(w_k,\ldots,w_{k+c}) = I_j(k,k+c) \rightarrow H_i) \quad \text{same} \quad \forall k,i,j$$

  - Same as in HMM for time invariance
- Context-Free
  - Probability of a subtree does not depend on words it does not dominates (spanning from *p* to *q*)

$$P(I_j(k,l) \rightarrow H_i \,|\, \text{outside} - \text{words}) = P(I_j(k,l) \rightarrow H_i)$$

- Ancestor-Free
  - Probability of a subtree does not depend on any derivation outside the subtree (spanning from *p* to *q*)

$$P(I_j(k,l) \rightarrow H_i \,|\, \text{outside} - \text{subtrees}) = P(I_j(k,l) \rightarrow H_i)$$

PCFG Computation and Inference

- Problem 1: Evaluation
  - How to compute *P(W|G)* efficiently?
  - Computing inside and outside probabilities
    - *inside-outside* algorithm for re-estimation
- Problem 2: Decoding
  - *Viterbi* algorithm: finding the most likely parse tree which also implies the most likely derivation sequence
  - Bayes Theorem: $\hat{t} = \arg\max_t P(t|W) = \arg\max_t P(W|t)P(t)$
- Problem 3: Parameter Estimation (Learning)
  - Given a set of observations *W*, determine the unknown values of the set of parameters (much more involved)

$$\theta = \{a_{ji} = P(I_j \rightarrow H_i): \ 1 \le i \le J_i, \ \ 1 \le j \le Q\}$$

- Countable State HMM (instead of finite state HMM)

Center of Signal and Image Processing
Georgia Institute of Technology
CSIP

Probability of a Rule

- Rule $r(i): A \rightarrow a$;
- Let R_A be the set of all rules $r(j)$, which have nonterminal A at the left-hand side;
- Then define probability distribution on R_A:

$$\sum_{r \in R_A} p(r) = 1,\ 0 \leq p(r) \leq 1$$

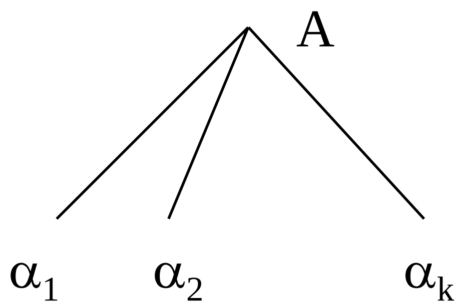
- Another point of view:

$p(a|A) = p(r)$, where $r = A \rightarrow a,\ a \in (N \cup T)^+$

Estimating Probability of a Rule

- MLE from a treebank ***following a PCFG grammar***
- Let $r = A \rightarrow a_1 a_2 \ldots a_k$:
  - $p(r) = c(r) / c(A)$
  - Counting rules *c(r):* how many instances appear in a treebank
  - Counting nonterminals *c(A):*

just count them in the treebank



Treebanks

- A collection of example parses by experts
- A commonly used treebank is the *Penn Treebank* *http://www.cis.upenn.edu/~treebank/*
- The induction problem is now that of extracting the grammatical knowledge that is implicit in the example parses
- Treebanks for other languages

Probability of a Derivation Tree

- Probabilistic CFG:
  - relations among a mother node & daughter nodes
  - in terms of expansion [rewrite,derivation] probability
  - define probability of a derivation (i.e. parse) tree:

$$P(T) = \Pi_{i=1..n} p(r(i))$$

- *r(i)* are all rules of the CFG used to generate the sentence *W* of which *T* is a parse
- Probability of a string $W = (w_1, w_2, \ldots, w_n)$?
- Non-trivial, because there may be many trees T_j as a result of a parsing *W*

Probability of a String with a D-Tree

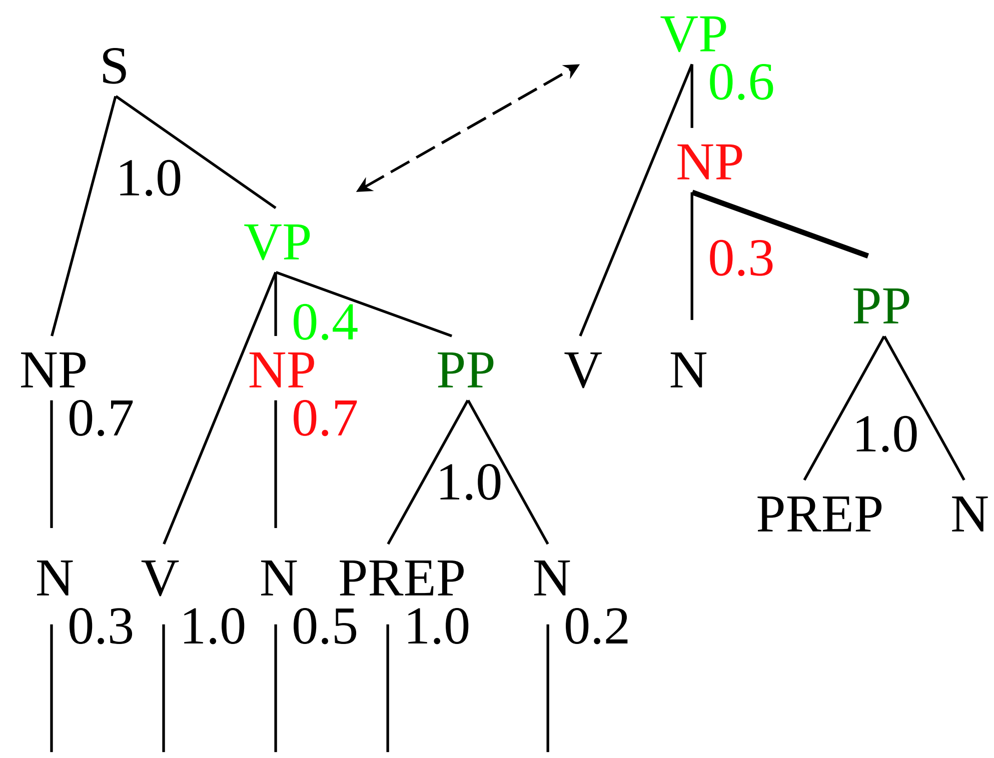
- Input string: W
- Parses: $\{d_j\}_{j=1..n}$ = Parse(W)

$$P(D) = \sum_{j=1..n} P(d_j)$$

- Hard to use the naive method

Example PCFG

- #1 S → NP VP 1.0
- #2 VP → V NP PP 0.4
- #3 VP → V NP 0.6
- #4 NP → N 0.7
- #5 NP → N PP 0.3
- #6 PP → PREP N 1.0
- #7 N → a_dog 0.3
- #8 N → a_cat 0.5
- #9 N → a_telescope 0.2
- #10 V → saw 1.0
- #11 PREP → with 1.0



P(a_dog saw a_cat with a_telescope) =

1×.7×.4×.3×.7×1×.5×1×1×.2 + ... ×.6... ×.3... = .00588 + .00378 = .00966

Computing String Probability

- a_dog saw a_cat with a_telescope
  1 2 3 4 5

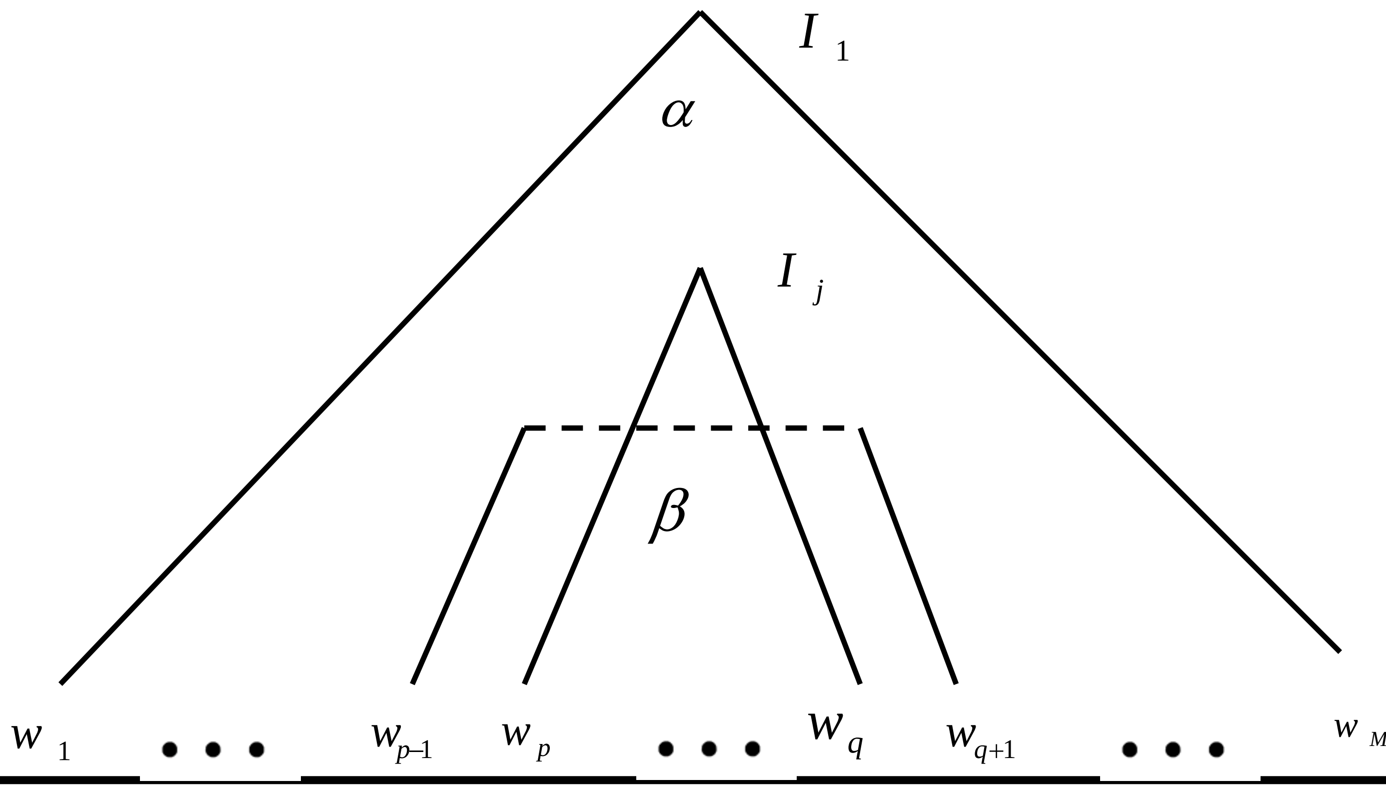
from\to	1	2	3	4	5
1	NP .21 N .3		S .441		S .00966
2		V 1	VP .21		VP .046
3			NP .35 N .5		NP .03
4				PREP 1	PP .2
5					N .2

- Create table *n* x *n* (*n* = length): cells might have more "lines"
- Initialize on diagonal, using $N \rightarrow a$ rules
- Recursively compute along diagonal towards upper right corner

Inside and Outside Probabilities

outside $\alpha_j(p,q) = P(w_1^{p-1}, I_j(p,q), w_{q+1}^M \mid G)$

inside $\beta_j(p,q) = P(w_p^q \mid I_j(p,q), G)$



Formula for Inside Probability

$$\beta_N(p,q) =$$

$$\Sigma_{A,B}\, \Sigma_{d=p..q-1}\, P(N \rightarrow A,B)\beta_A(p,d)\beta_B(d+1,q)$$

- assuming the grammar *G* has rules of the form

N → *w* (terminal string only)

N → *A B* (two nonterminals)

- only (Chomsky Normal Form, or CRF)

Computing Inside Probability

- Terminal-word derivation

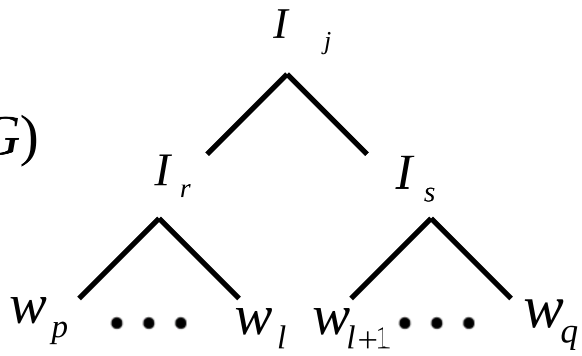
$$\beta_j(k,k) = P(w_k \mid I_j(k,k), G) = P(I_j \rightarrow w_k \mid G)$$

- Root sentence derivation

$$P(w_1^M \mid G) = P(I_1 \rightarrow w_1^M \mid G) = P(W \mid I_1(1,M), G) = \beta_1(1,M)$$

- Inside Algorithm (Bottom-Up Induction)

$$\beta_j(p,q) = P(w_p^q \mid I_j(p,q), G)$$
$$= \sum_{r,s} \sum_{l=p}^{q-1} P(w_p^l, I_r(p,l), w_{l+1}^q, I_s(l+1,q) \mid I_j(p,q), G)$$
$$= \sum_{r,s} \sum_{l=p}^{q-1} P(I_j \rightarrow I_r I_s) \beta_r(p,l) \beta_s(l+1,q)$$



Computing Outside Probability

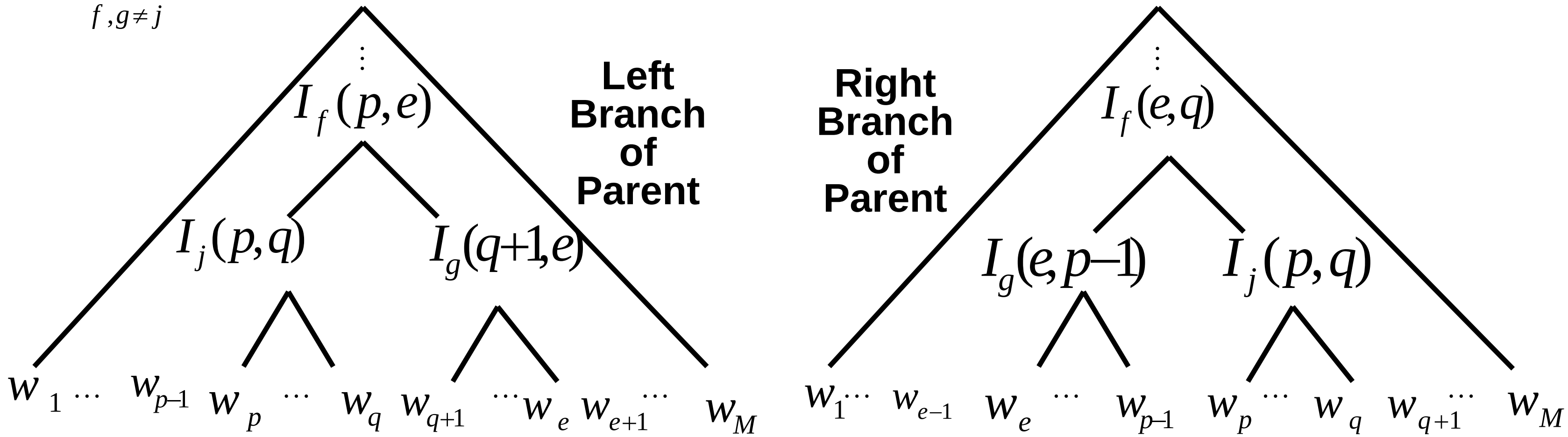
- Terminal derivation

$$P(w_1^M \mid G) = \sum_j P(w_1^{k-1}, w_k, w_{k+1}^M, I_j(k,k) \mid G) = \sum_j \alpha_j(k,k) P(I_j \rightarrow w_k)$$

- Root sentence derivation $\alpha_1(1,M) = 1 \quad \alpha_j(1,M) = 0 \quad j \neq 1$
- Outside Algorithm (Top-Down Induction)

$$\alpha_j(p,q) = [\sum_{f,g} \sum_{e=q+1}^{M} P(w_1^{p-1}, w_{q+1}^M, I_f(p,e), I_j(p,q), I_g(q+1,e))]$$

$$+[\sum_{f,g \neq j} \sum_{e=1}^{p-1} P(w_1^{p-1}, w_{q+1}^M, I_f(e,q), I_g(e,p-1), I_j(p,q))]$$



Computing Outside Prob. (Cont.)

- Outside Algorithm (Top-Down Induction)

$$\alpha_j(p,q) = [\sum_{f,g} \sum_{e=q+1}^{M} \alpha_f(p,e) P(I_f \to I_j I_g) \beta_g(q+1,e)]$$

$$+ [\sum_{f,g \neq j} \sum_{e=1}^{p-1} \alpha_f(e,q) P(I_f \to I_g I_j) \beta_g(e,p-1)]$$

- Inside-Outside Probability Product

$$\alpha_j(p,q)\beta_j(p,q) = P(w_1^M, I_j(p,q) \mid G)$$

$$= P(w_1^{p-1}, I_j(p,q), w_{q+1}^M \mid G) * P(w_p^q \mid I_j(p,q), G)$$

- Is there a bracket from position *p* to *q* ?

$$P(w_1^M, I(p,q) \mid G) = \sum_j \alpha_j(p,q)\beta_j(p,q)$$

- Pre-terminal (Non-terminal parent of a terminal)

$$P(w_1^M, I(k,k) \mid G) = \sum_j \alpha_j(k,k)\beta_j(k,k) = \alpha_1(1,M)\beta_1(1,M)$$

Decoding the Most Likely Parse

- Computing Optimal Partial Path Scores
  - remember DP recursion (Principle of Optimality) !!
- Initialization $\delta_i(k,k) = P(I_j \rightarrow w_k)$
- DP-Recursion and Bookkeeping

$$\delta_j(p,q) = \max_{1 \le r,s \le N,\, p \le e < q} [\delta_r(p,e)\delta_s(e+1,q)P(I_j \rightarrow I_r I_s)]$$

$$\psi_j(p,q) = \arg\max_{(r,s,e)} [\delta_r(p,e)\delta_s(e+1,q)P(I_j \rightarrow I_r I_s)]$$

- Termination (M-level Parse Tree)

$$P_{\max} = \max_{1 \le j \le N} \delta_j(1,M) \quad \text{and} \quad \hat{d}_M = \arg\max_{1 \le j \le N} \psi_j(1,M)$$

- Traceback (left/right branches and break point)

$$\hat{d}_{m-1} = \psi_{\hat{d}_m}(p,q) = (\hat{r}_m, \hat{s}_m, \hat{e}_m) \quad m = M, M\text{-}1, \ldots, 2$$

- "Optimal" Derivation Sequence: $\hat{\mathrm{t}} = (\hat{d}_1, \ldots, \hat{d}_M)$

Center of Signal and Image Processing
Georgia Institute of Technology
CSIP

PCFG Parameter Estimation

- Counting: for each training sentence *W(i)*

$$f_i(p,q,j,r,s)=\sum_{e=p}^{q-1}\alpha_j(p,q)P(I_j\rightarrow I_rI_s)\beta_r(p,e)\beta(e+1,q)$$

$$g_i(l,j,k)=\alpha_j(l,l)P(w_l=w_k)\beta_j(l,l) \qquad h_i(p,q,j)=\alpha_j(p,q)\beta_j(p,q)$$

- ML Re-estimation of PCFG Parameters

$$\hat{P}(I_j\rightarrow I_rI_s)=\frac{\sum_{i=1}^{Q}\sum_{p=1}^{M(i)-1}\sum_{q=p+1}^{M(i)}[f_i(p,q,j,r,s)/P(I_1\rightarrow W(i))]}{\sum_{i=1}^{Q}\sum_{p=1}^{M(i)}\sum_{q=p}^{M(i)}[h_i(p,q,j)/P(I_1\rightarrow W(i))]}$$

$$\hat{P}(I_j\rightarrow w_k)=\frac{\sum_{i=1}^{Q}\sum_{l=1}^{M(i)}[g_i(l,j,k)/P(I_1\rightarrow W(i))]}{\sum_{i=1}^{Q}\sum_{p=1}^{M(i)}\sum_{q=p}^{M(i)}[h_i(p,q,j)/P(I_1\rightarrow W(i))]}$$

- Solving the fixed point problem: EM algorithm
  - E-step: compute new counts with old parameter estimates
  - M-step: re-estimate parameters with new counts

Some Issues Before Moving On

- **Problems with PCFG Estimation**
  - many unsolved research issues: *<u>less studied, more rewards</u>*
  - sizes of *A* and *I* often unknown: O(M*M*M*Q*Q*Q)
  - too little data to estimate too many parameters
  - greater *A* and *I* imply more estimation & storage problem
  - techniques in search, N-gram and HMM can be extended
- **Parsing for Disambiguation and Understanding?**
  - probabilities for determining the sentence
  - probabilities for speedier parsing (pruning efficiency)
  - probabilities for choosing between parses (ranking/scoring)
- **Labeled Corpus for Learning - Treebank**
  - chunking (bracketing): the first step to studying parsing
  - Penn Treebank: widely used, large size; other languages?

Summary

- Today's Class
  - Probabilistic Context Free Grammar
- Next Classes
  - Statistical Parsing
  - Lab 6 assigned
  - Project monitoring
- Reading Assignments
  - Manning and Schutze, Chapters 11-12