Linguistics 696f
Hammond
Spring '03

PCFG algorithms

A. Overview

(1)
 a. PCFGs
 b. CNF
 c. Chart parsing
 d. Forward probabilities
 e. Backward probabilities
 f. Inside-outside algorithm

B. What is a PCFG?

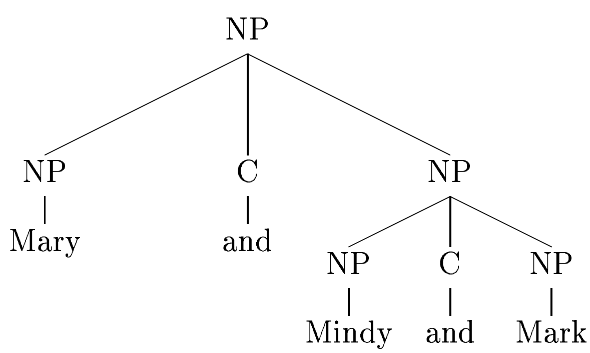
(2) Sentence probability: $\mathrm{p}(s) = \sum_j \mathrm{p}(t_j)\mathrm{p}(s|t_j)$

(3)

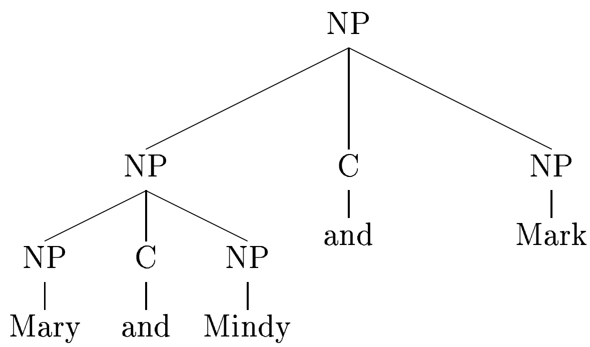
NP	$\rightarrow$	NP C NP	.4
NP	$\rightarrow$	Mary	.3
NP	$\rightarrow$	Mindy	.2
NP	$\rightarrow$	Mark	.1
C	$\rightarrow$	and	1

(4) The probability of each parse is: $.3 \times .2 \times .1 \times 1 \times 1 \times .4 \times .4 = .00096$. The overall probability of the string is $.00096 + .00096 = .00192$.

(5)



(6)



C. Chomsky-normal form

(7) Any context-free grammar can be expressed as in Chomsky-normal form (CNF) without any change in expressive power.

(8) Chomsky-Normal Form
a. $A \rightarrow BC$, where A, B, and C are non-terminal symbols.
b. $A \rightarrow a$, where A is a non-terminal and a is a single terminal.

(9) Demonstrating the inside-outside algorithm is much easier with a CFG in CNF.

D. Chart parsing

(10) How do we find whether a sentence is legal with respect to some CFG?

(11) This is a hard problem because, in principle, we must keep track of every possible application of every rewrite rule as we proceed from left to right.

(12) *The horse raced past the barn fell.*

(13) Chart parsing (dynamic programming) is an efficient way to do this.

(14) Chart parsing
Keep track of *all* successful rewrite rule applications in a chart as you parse left to right.

(15)

<table><tr><td>*Mary*</td></tr><tr><td>NP</td></tr></table>

$\rightarrow$

<table><tr><td>*Mary*</td><td>*and*</td></tr><tr><td>NP</td><td>C</td></tr></table>

$\rightarrow$

<table><tr><td>*Mary*</td><td>*and*</td><td>*Mindy*</td></tr><tr><td>NP</td><td>C</td><td>NP</td></tr><tr><td colspan="3">NP</td></tr></table>

$\rightarrow$

$\rightarrow$

<table><tr><td>*Mary*</td><td>*and*</td><td>*Mindy*</td><td>*and*</td></tr><tr><td>NP</td><td>C</td><td>NP</td><td>C</td></tr><tr><td colspan="3">NP</td><td></td></tr></table>

$\rightarrow$

$\rightarrow$

<table><tr><td>*Mary*</td><td>*and*</td><td>*Mindy*</td><td>*and*</td><td>*Mark*</td></tr><tr><td>NP</td><td>C</td><td>NP</td><td>C</td><td>NP</td></tr><tr><td colspan="3">NP</td><td colspan="2"></td></tr><tr><td colspan="2"></td><td colspan="3">NP</td></tr><tr><td colspan="5">NP</td></tr></table>

(16) Why is this efficient?:

a. We do *not* keep track of all complete parses.

b. Once we know a rule can succeed, we record that success for all future parses.

E. Inside and outside probabilities

(17) Backward probability = "inside" probability

$\beta_j(k,l) \stackrel{\text{def}}{=} \mathrm{p}(w_{k,l}|N^j_{k,l})$

(18) Base case for inside probability for PCFG in CNF

$\beta_j(k,k) = \mathrm{p}(w_k|N^j_{k,k}) = \mathrm{p}(N^j \rightarrow w_k)$

(19) Recursive case for inside probability for PCFG in CNF

$\beta_j(k,l) \stackrel{\text{def}}{=} \sum_{p,q,m}\ \mathrm{p}(N^j \rightarrow N^p N^q)\ \beta_p(k,m)\ \beta_q(m+1,l)$

(20) Forward probability = "outside" probability

$\alpha_j(k,l) \stackrel{\text{def}}{=} \mathrm{p}(w_{1,k-1}, N^j_{k,l}, w_{l+1,n})$

(21) Base case for outside probability for PCFG in CNF

$\alpha_1(1,n) = \mathrm{p}(N^1_{1,n}) = 1$

(22) Recursive case for outside probability for PCFG in CNF

$$\begin{aligned}\alpha_j(k,l) \stackrel{\text{def}}{=} & \sum_{h,p,q} \alpha_p(h,l)\ \mathrm{p}(N^p \rightarrow N^q N^j)\ \beta_q(h,k-1) \quad + \\ & \sum_{m,p,q} \alpha_p(k,m)\ \mathrm{p}(N^p \rightarrow N^j N^q)\ \beta_q(l+1,m)\end{aligned}$$

F. Training a PCFG

(23) $|N^j \to N^p N^q| \stackrel{\text{def}}{=} \frac{1}{\mathrm{p}(w_{1,n})} \sum_{k,l,m} \ \alpha_j(k,l)\ \mathrm{p}(N^j \to N^p N^q)\ \beta_p(k,m)\ \beta_q(m+1,l)$

References

CHARNIAK, EUGENE. 1993. *Statistical Language Learning*. Cambridge: MIT Press.

HAMMOND, MICHAEL, 2003. Statistical natural language processing. U. of Arizona.

HOPCROFT, J.E., & J.D. ULLMAN. 1979. *Introduction to Automata Theory, Languages, and Computation*. Reading: Addison-Wesley.

SUPPES, PATRICK. 1970. Probabilistic grammars for natural languages. *Synthese* 22.95–116.