Probabilistic Context Free Grammars

Lecture #16

Introduction to Natural Language Processing
CMPSCI 585, Spring 2004

Andrew McCallum

(including slides from Jason Eisner)

Ambiguity in Parsing

- Time flies like an arrow.
- Fruit flies like a banana.

- I saw the man with the telescope.

How to solve this combinatorial explosion of ambiguity?

1. First try parsing without any weird rules, throwing them in only if needed.
2. Better: every rule has a weight.
   A tree's weight is total weight of all its rules.
   Pick the overall "lightest" parse of sentence.
3. Can we pick the weights automatically?
   We'll get to this later …

CYK Parser

Input: A string of words, grammar in CNF
Output: yes/no
Data structure: n x n table
  rows labeled 0 to n-1, columns 1 to n
  cell (i,j) lists constituents spanning i,j

For each i from 1 to n
  Add to (i-1,i) all Nonterminals that could produce the word at (i-1,i)

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3				
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

CYK Parser

For **width** from 2 to n

For **start** from 0 to n-width

Define **end** to be **start+width**

For **mid** from **start+1** to **end-1**

For every constituent in (**start, mid**)

For every constituent in (**mid,end**)

For all ways of combining them (if any)

Add the resulting constituent to (**start,end**).

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3				
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		PP 12
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			NP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			NP 18 S 21
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP

2 VP → VP PP
1 NP → Det N

2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP

2 VP → VP PP
1 NP → Det N

2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

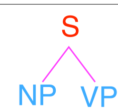
Follow backpointers …

S

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP



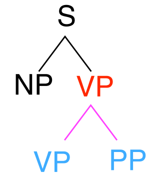
time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

time 1 flies 2 like 3 an 4 arrow 5

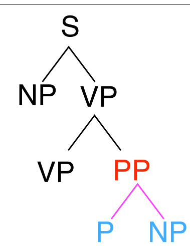
	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8



1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

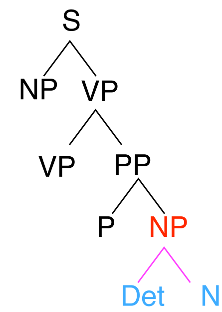
time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8



1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8



1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Which entries do we need?

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Which entries do we need?

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Not worth keeping …

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

… since it just breeds worse options

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Keep only best-in-class!

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

inferior stock

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Keep only best-in-class!

(and backpointers so you can recover parse)

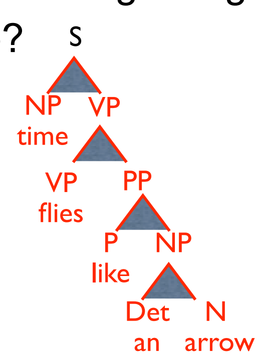
time 1 flies 2 like 3 an 4 arrow 5

	NP 3 Vst 3	NP 10 S 8			NP 24 S 22
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Probabilistic Trees

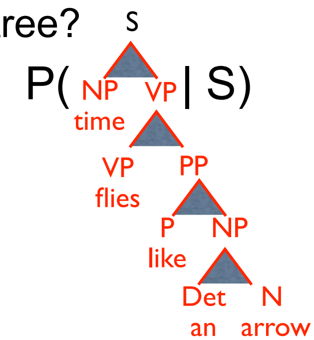
- Instead of lightest weight tree, take highest probability tree
- Given any tree, your assignment generator would have some probability of producing it!
- Just like using n-grams to choose among strings …
- What is the probability of this tree?



Probabilistic Trees

- Instead of lightest weight tree, take highest probability tree
- Given any tree, your assignment generator would have some probability of producing it!
- Just like using n-grams to choose among strings …
- What is the probability of this tree?
-
- You rolled a lot

of independent dice...



Chain rule: One word at a time

p(time flies like an arrow)
= p(time)
* p(flies | time)
* p(like | time flies)
* p(an | time flies like)
* p(arrow | time flies like an)

Chain rule + backoff
(to get trigram model)

p(time flies like an arrow)
= p(time)
* p(flies | time)
* p(like | time flies)
* p(an | time flies like)
* p(arrow | time flies like an)

Chain rule – written differently

p(time flies like an arrow)
= p(time)
* p(time flies | time)
* p(time flies like | time flies)
* p(time flies like an | time flies like)
* p(time flies like an arrow | time flies like an)

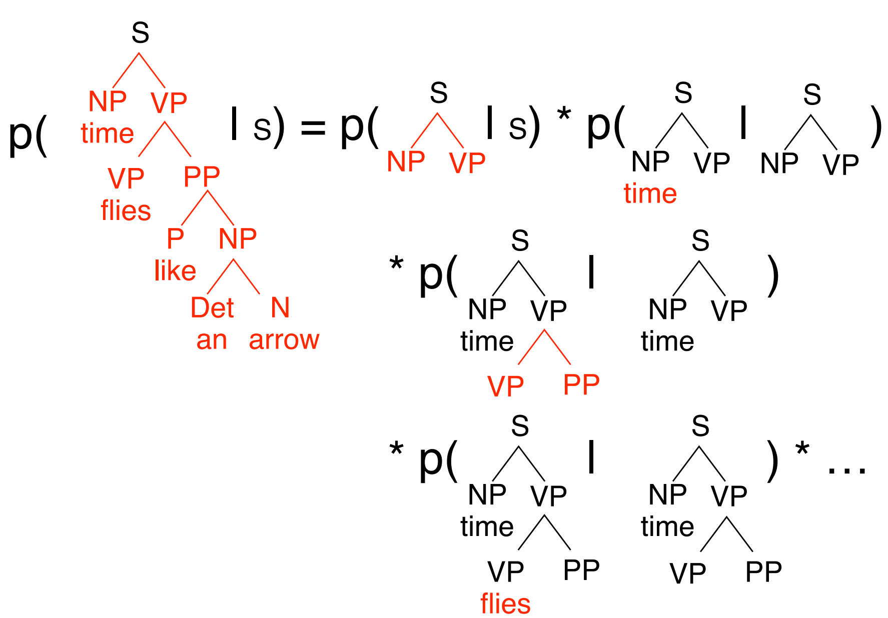
Proof: p(x,y | x) = p(x | x) * p(y | x, x) = 1 * p(y | x)

Chain rule + backoff

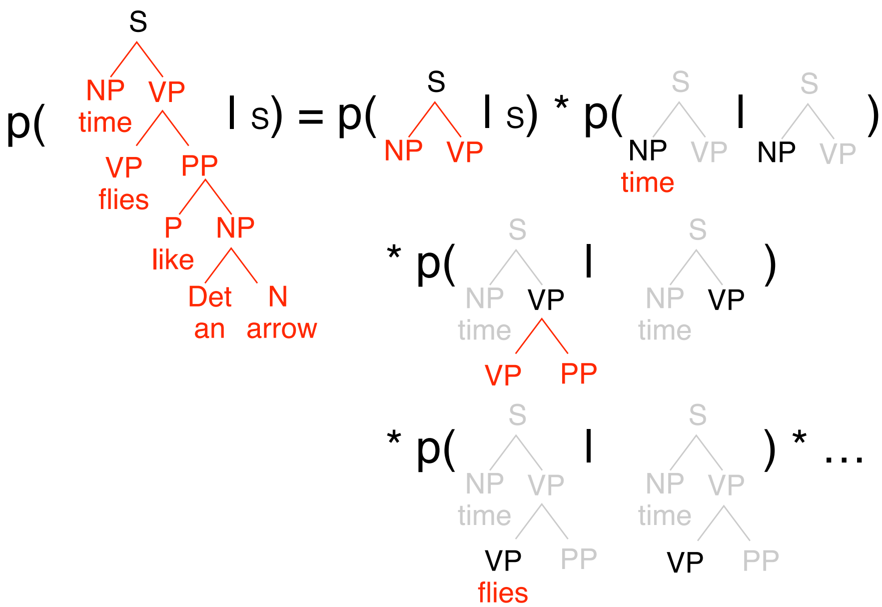
p(time flies like an arrow)
 = p(time)
 * p(time flies | time)
 * p(time flies like | time flies)
 * p(time flies like an | time flies like)
 * p(time flies like an arrow | time flies like an)

Proof: p(x,y | x) = p(x | x) * p(y | x, x) = 1 * p(y | x)

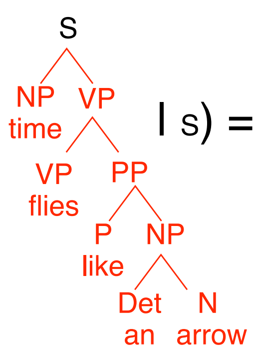
Chain rule: One node at a time



Chain rule + backoff



Simplified notation

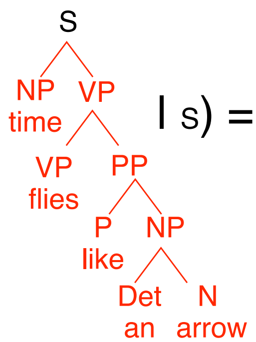


$p($ tree $\mid \text{S}) = p(\text{S} \rightarrow \text{NP VP} \mid \text{S}) * p(\text{NP} \rightarrow \text{flies} \mid \text{NP})$

$* \; p(\text{VP} \rightarrow \text{VP NP} \mid \text{VP})$

$* \; p(\text{VP} \rightarrow \text{flies} \mid \text{VP}) * \ldots$

Already have a CKY alg for weights …

w(

| S) = w(S → NP VP) + w(NP → flies | NP)

+ w(VP → VP NP)

+ w(VP → flies) + …

Just let w(X → Y Z) = -log p(X → Y Z | X)

Then lightest tree has highest prob 49

---

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

2^{-8}

multiply to get 2^{-22}

2^{-12}

2^{-2}

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Need only best-in-class to get best parse

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S 8 S 13			NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

2^{-13}

2^{-8}

multiply to get 2^{-22}

2^{-12}

2^{-2}

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Why probabilities not weights?

- We just saw probabilities are really just a special case of weights …
- … *but* we can estimate them from training data by counting and smoothing! Use all of our lovely probability theory machinery!

Probabilistic Context Free Grammars (PCFGs)

A PCFG G consists of the usual parts of a CFG

- A set of terminals, $\{w^k\}, k = 1, \ldots, V$
- A set of nonterminals, $\{N^i\}, i = 1, \ldots, n$
- A designated start symbol, N^1
- A set of rules, $\{N^i \to \zeta^j\}$, (where ζ^j is a sequence of terminals and nonterminals)

and

- A corresponding set of probabilities on rules such that:

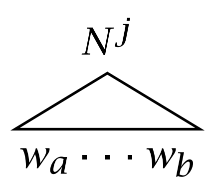
$$\forall i \quad \sum_j P(N^i \to \zeta^j) = 1$$

PCFG notation

Sentence: sequence of words $w_1 \cdots w_m$

w_{ab}: the subsequence $w_a \cdots w_b$

N^i_{ab}: nonterminal N^i dominates $w_a \cdots w_b$



$N^i \overset{*}{\Longrightarrow} \zeta$: Repeated derivation from N^i gives ζ.

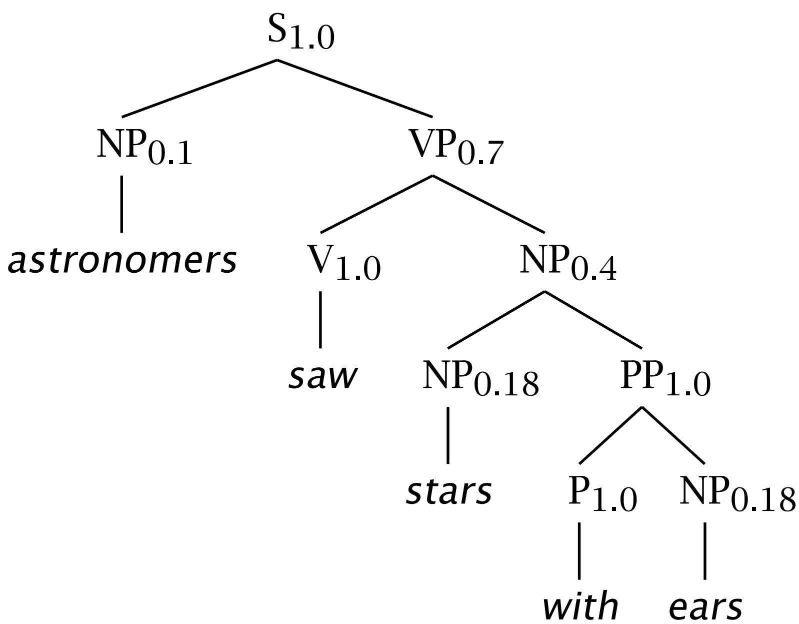
PCFG probability of a string

$$P(w_{1n}) = \sum_t P(w_{1n}, t) \quad t \text{ a parse of } w_{1n}$$
$$= \sum_{\{t:\text{yield(t)}=\text{w}_{1\text{n}}\}} P(t)$$

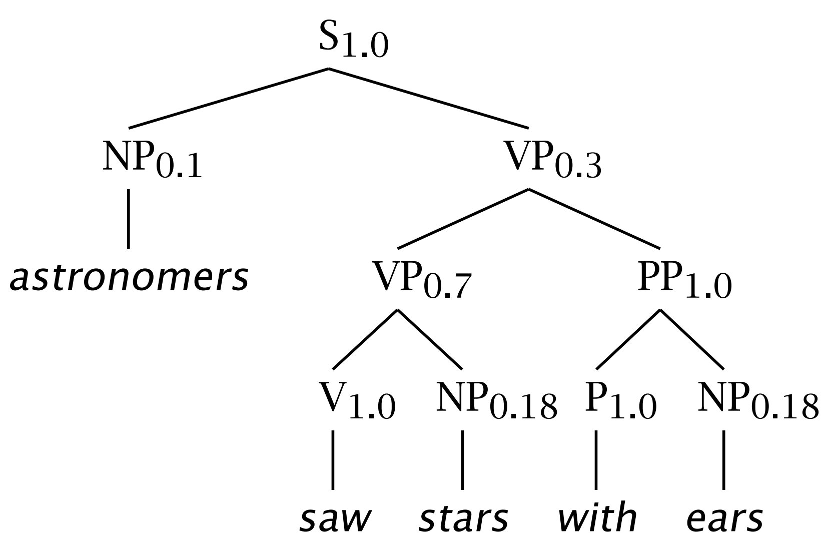
A simple PCFG (in CNF)

S → NP VP	1.0	NP → NP PP	0.4
PP → P NP	1.0	NP → *astronomers*	0.1
VP → V NP	0.7	NP → *ears*	0.18
VP → VP PP	0.3	NP → *saw*	0.04
P → *with*	1.0	NP → *stars*	0.18
V → *saw*	1.0	NP → *telescopes*	0.1

t_1:


S1.0
NP0.1
VP0.7
astronomers
V1.0
NP0.4
saw
NP0.18
PP1.0
stars
P1.0
NP0.18
with
ears

t_2:


S1.0
NP0.1
VP0.3
astronomers
VP0.7
PP1.0
V1.0
NP0.18
P1.0
NP0.18
saw
stars
with
ears

The two parse trees' probabilities and the sentence probability

$$\begin{aligned}
P(t_1) &= 1.0 \times 0.1 \times 0.7 \times 1.0 \times 0.4 \\
&\quad \times 0.18 \times 1.0 \times 1.0 \times 0.18 \\
&= 0.0009072 \\
P(t_2) &= 1.0 \times 0.1 \times 0.3 \times 0.7 \times 1.0 \\
&\quad \times 0.18 \times 1.0 \times 1.0 \times 0.18 \\
&= 0.0006804 \\
P(w_{15}) &= P(t_1) + P(t_2) = 0.0015876
\end{aligned}$$

Assumptions of PCFGs

1. Place invariance (like time invariance in HMM):

$$\forall k \quad P(N^j_{k(k+c)} \rightarrow \zeta) \text{ is the same}$$

2. Context-free:

$$P(N^j_{kl} \rightarrow \zeta | \text{words outside } w_k \ldots w_l) = P(N^j_{kl} \rightarrow \zeta)$$

3. Ancestor-free:

$$P(N^j_{kl} \rightarrow \zeta | \text{ancestor nodes of } N^j_{kl}) = P(N^j_{kl} \rightarrow \zeta)$$

The sufficient statistics of a PCFG are thus simply counts of how often different local tree configurations occurred (= counts of which grammar rules were applied).

Some features of PCFGs

Reasons to use a PCFG, and some idea of their limitations:

- Partial solution for grammar ambiguity: a PCFG gives some idea of the plausibility of a sentence.
- But, in the simple case, not a very good idea, as independence assumptions are two strong (e.g., not lexicalized).
- Gives a probabilistic language model for English.
- In the simple case, a PCFG is a worse language model for English than a trigram model.
- Better for grammar induction (Gold 1967 vs. Horning 1969)
- Robustness. (Admit everything with low probability.)

Some features of PCFGs

- A PCFG encodes certain biases, e.g., that smaller trees are normally more probable.
- One can hope to combine the strengths of a PCFG and a trigram model.

We'll look at simple PCFGs first. They have certain inadequacies, but we'll see that most of the state-of-the-art probabilistic parsers are fundamentally PCFG models, just with various enrichments to the grammar

A slightly different task

- Been asking: What is probability of generating a given *tree*?
  - To pick tree with highest prob: useful in parsing.
-
- But could also ask: What is probability of generating a given *string* with the generator?
  - To pick string with highest prob: useful in speech recognition, as substitute for an n-gram model.
    - ("Put the file in the folder" vs. "Put the file and the folder")
  - To get prob of generating string, must add up probabilities of all trees for the string …

Could just add up the parse probabilities

time 1 flies 2 like 3 an 4 arrow 5

	1	2	3	4	5
0	NP 3 Vst 3	NP 10 S 8 S 13		2^{-22} 2^{-27} 2^{-27} 2^{-22} 2^{-27}	NP 24 S 22 S 27 NP 24 S 27 S 22 S 27
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 12 VP 16
3				Det 1	NP 10
4					N 8

oops, back to finding exponentially many parses

1 S → NP VP
6 S → Vst NP
2 S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Any more efficient way?

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S 2^{-8} S 2^{-13}			NP 24 S 22 S 27 NP 24 S 27 S 2^{-22} S 2^{-27}
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 2^{-12} VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2^{-2} S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Add as we go … (the "inside algorithm")

time 1 flies 2 like 3 an 4 arrow 5

0	NP 3 Vst 3	NP 10 S $2^{-8}+2^{-13}$			NP 24 S 22 S 27 NP 24 S 27 S 2^{-22} $+2^{-27}$
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 2^{-12} VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2^{-2} S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Add as we go … (the “inside algorithm”)

	time 1	flies 2	like 3	an 4	arrow 5
0	NP 3 Vst 3	NP 10 S $2^{-8}+2^{-13}$			NP 2^{-22} $+2^{-27}$ S 2^{-22} $+2^{-27}$ $+2^{-27}$ $+2^{-22}$ $+2^{-27}$
1		NP 4 VP 4			NP 18 S 21 VP 18
2			P 2 V 5		PP 2^{-12} VP 16
3				Det 1	NP 10
4					N 8

1 S → NP VP
6 S → Vst NP
2^{-2} S → S PP
1 VP → V NP
2 VP → VP PP
1 NP → Det N
2 NP → NP PP
3 NP → NP NP
0 PP → P NP

Inside and Outside Probabilities

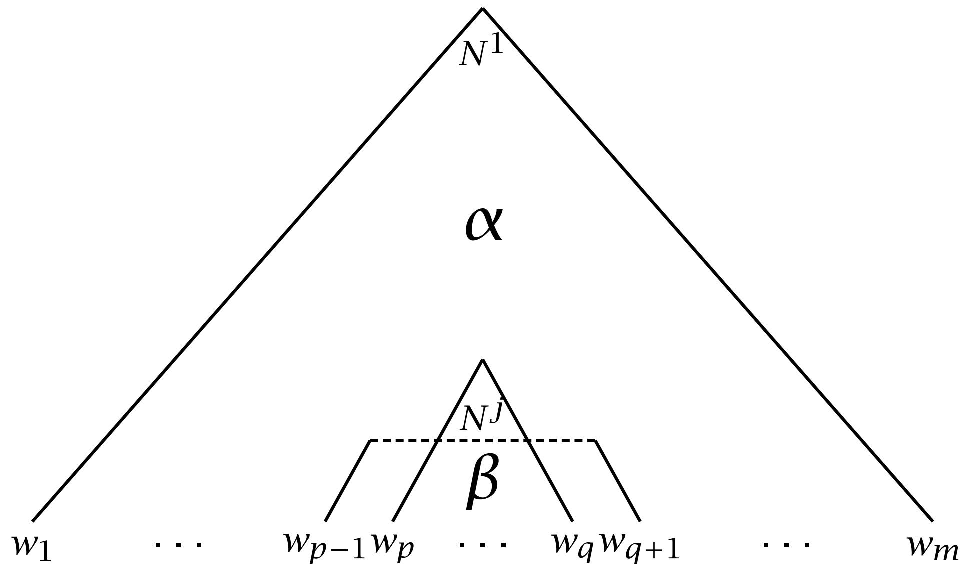
Probability of all possible rule re-writes for generating words inside position p to q, given that non-terminal j exactly spans p to q.

$$\text{Inside} = \beta_j(p,q) = P(w_{pq} | N^j_{pq}, G)$$

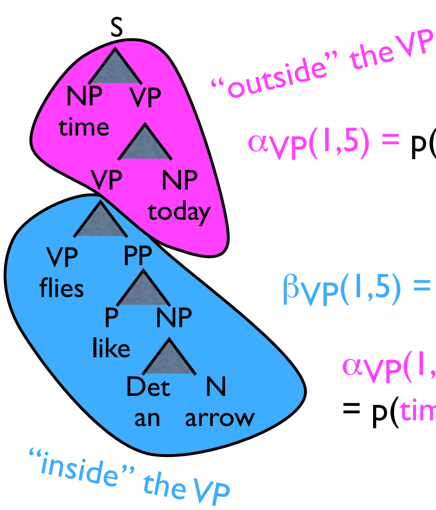
Probability of all possible rule re-writes for generating words *outside* position p to q, *and* that non-terminal j exactly spans p to q.

$$\text{Outside} = \alpha_j(p,q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m} | G)$$

Inside and Outside Probabilities



Inside & Outside Probabilities



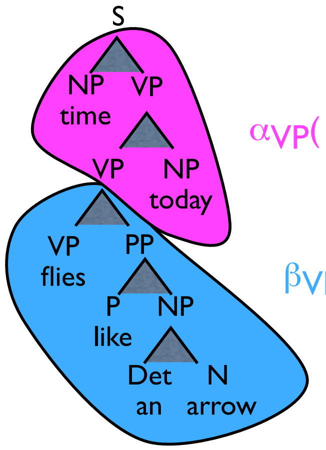
$\alpha_{VP}(1,5)$ = p(time VP today | S)

$\beta_{VP}(1,5)$ = p(flies like an arrow | VP)

$\alpha_{VP}(1,5) * \beta_{VP}(1,5)$
= p(time [VP flies like an arrow] today | S)

Andrew McCallum, UMass

Inside & Outside Probabilities



$\alpha_{VP}(1,5)$ = p(time VP today | S)

$\beta_{VP}(1,5)$ = p(flies like an arrow | VP)

$\alpha_{VP}(1,5) * \beta_{VP}(1,5) \ / \ \beta_S(0,6)$

$= \dfrac{\text{p(time flies like an arrow today \& VP(1,5) | S)}}{\text{p(time flies like an arrow today | S)}}$

= p(VP(1,5) | time flies like an arrow today, S)

So $\alpha_{VP}(1,5) * \beta_{VP}(1,5) \ / \ \beta_S(0,6)$
is probability that there is a VP here,
given all of the observed data (words)

Probability of a string

Inside probability

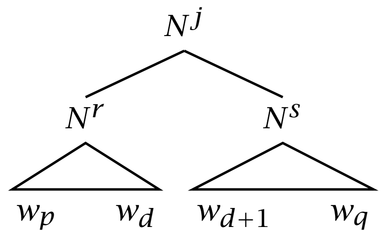
$$\begin{aligned} P(w_{1m}|G) &= P(N^1 \Rightarrow w_{1m}|G) \\ &= P(w_{1m}, N^1_{1m}, G) = \beta_1(1,m) \end{aligned}$$

Base case: We want to find $\beta_j(k,k)$ (the probability of a rule $N^j \rightarrow w_k$):

$$\begin{aligned} \beta_j(k,k) &= P(w_k|N^j_{kk}, G) \\ &= P(N^j \rightarrow w_k|G) \end{aligned}$$

Probability of a string

Induction: We want to find $\beta_j(p,q)$, for $p < q$. As this is the inductive step using a Chomsky Normal Form grammar, the first rule must be of the form $N^j \rightarrow N^r \quad N^s$, so we can proceed by induction, dividing the string in two in various places and summing the result:



These inside probabilities can be calculated bottom up.

For all j,

$$
\begin{aligned}
\beta_j(p,q) &= P(w_{pq}|N^j_{pq}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(w_{pd}, N^r_{pd}, w_{(d+1)q}, N^s_{(d+1)q}|N^j_{pq}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^r_{pd}, N^s_{(d+1)q}|N^j_{pq}, G) \\
&\qquad P(w_{pd}|N^j_{pq}, N^r_{pd}, N^s_{(d+1)q}, G) \\
&\qquad P(w_{(d+1)q}|N^j_{pq}, N^r_{pd}, N^s_{(d+1)q}, w_{pd}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^r_{pd}, N^s_{(d+1)q}|N^j_{pq}, G) \\
&\qquad P(w_{pd}|N^r_{pd}, G) P(w_{(d+1)q}|N^s_{(d+1)q}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s)\beta_r(p,d)\beta_s(d+1,q)
\end{aligned}
$$

Inside probabilities as CYK

	1	2	3	4	5
1	$\beta_{NP} =$ 0.1		$\beta_S =$ 0.0126		$\beta_S =$ 0.0015876
2		$\beta_{NP} =$ 0.04 $\beta_V =$ 1.0	$\beta_{VP} =$ 0.126		$\beta_{VP} =$ 0.015876
3			$\beta_{NP} =$ 0.18		$\beta_{NP} =$ 0.01296
4				$\beta_P =$ 1.0	$\beta_{PP} =$ 0.18
5					$\beta_{NP} =$ 0.18
	astronomers	*saw*	*stars*	*with*	*ears*

Outside probabilities

Probability of a string: For any k, $1 \le k \le m$,

$$\begin{aligned} P(w_{1m}|G) &= \sum_j P(w_{1(k-1)}, w_k, w_{(k+1)m}, N^j_{kk}|G) \\ &= \sum_j P(w_{1(k-1)}, N^j_{kk}, w_{(k+1)m}|G) \\ &\qquad \times P(w_k|w_{1(k-1)}, N^j_{kk}, w_{(k+1)n}, G) \\ &= \sum_j \alpha_j(k,k) P(N^j \to w_k) \end{aligned}$$

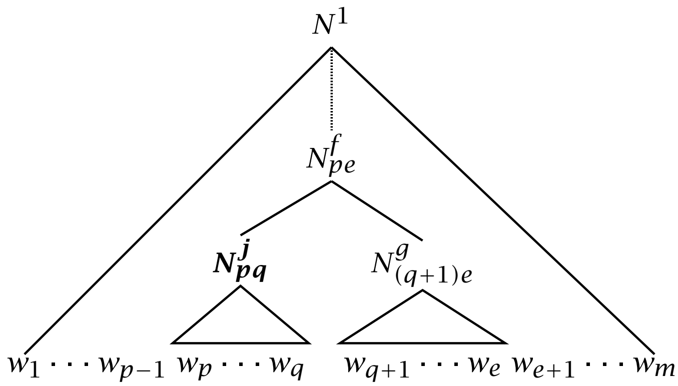
Inductive (DP) calculation: One calculates the outside probabilities top down (after determining the inside probabilities).

Outside probabilities

Base Case:

$$\alpha_1(1,m) = 1$$
$$\alpha_j(1,m) = 0, \text{ for } j \neq 1$$

Inductive Case: it's either a left or right branch – we will some over both possibilities and calculate using outside *and* inside probabilities



Outside probabilities, Inductive case

$$\begin{aligned}
\alpha_j(p,q) &= \Big[\sum_{f,g}\sum_{e=q+1}^{m} P(w_{1(p-1)}, w_{(q+1)m}, N^f_{pe}, N^j_{pq}, N^g_{(q+1)e})\Big] \\
&\quad + \Big[\sum_{f,g}\sum_{e=1}^{p-1} P(w_{1(p-1)}, w_{(q+1)m}, N^f_{eq}, N^g_{e(p-1)}, N^j_{pq})\Big] \\
&= \Big[\sum_{f,g}\sum_{e=q+1}^{m} P(w_{1(p-1)}, w_{(e+1)m}, N^f_{pe}) P(N^j_{pq}, N^g_{(q+1)e} | N^f_{pe}) \\
&\quad \times P(w_{(q+1)e} | N^g_{(q+1)e})\Big] + \Big[\sum_{f,g}\sum_{e=1}^{p-1} P(w_{1(e-1)}, w_{(q+1)m}, N^f_{eq}) \\
&\quad \times P(N^g_{e(p-1)}, N^j_{pq} | N^f_{eq}) P(w_{e(p-1)} | N^g_{e(p-1)})\Big] \\
&= \Big[\sum_{f,g}\sum_{e=q+1}^{m} \alpha_f(p,e) P(N^f \to N^j\ N^g) \beta_g(q+1,e)\Big] \\
&\quad + \Big[\sum_{f,g}\sum_{e=1}^{p-1} \alpha_f(e,q) P(N^f \to N^g\ N^j) \beta_g(e,p-1)\Big]
\end{aligned}$$

Probability that a rule is used

As with a HMM, we can form a product of the inside and outside probabilities. This time:

$$\alpha_j(p,q)\beta_j(p,q)$$
$$= P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m}|G)P(w_{pq}|N^j_{pq}, G)$$
$$= P(w_{1m}, N^j_{pq}|G)$$

$$P(N^j_{pq}|w_{1m}, G) = \frac{P(N^j_{pq}|w_{1m}, G)}{P(w_{1m}|G)} = \frac{\alpha_j(p,q)\beta_j(p,q)}{\beta_1(1,m)}$$

This is an "expected count" for the number of times this rule occurred.

Overall probability of a node existing

As with a HMM, we can form a product of the inside and outside probabilities. This time:

$$\alpha_j(p,q)\beta_j(p,q)$$
$$= P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m}|G)P(w_{pq}|N^j_{pq}, G)$$
$$= P(w_{1m}, N^j_{pq}|G)$$

Therefore,

$$p(w_{1m}, N_{pq}|G) = \sum_j \alpha_j(p,q)\beta_j(p,q)$$

Just in the cases of the root node and the preterminals, we know there will always be some such constituent.

Learning PCFGs (1)

- We would like to calculate how often each rule is used:

$$\hat{P}(N^j \to \zeta) = \frac{C(N^j \to \zeta)}{\sum_{\gamma} C(N^j \to \gamma)}$$

- If we have labeled data, we count and find out
- Relative frequency again gives maximum likelihood probability estimates
- This is the motivation for building *Treebanks* of hand-parsed sentences

Learning PCFGs (2) Inside-Outside

- Otherwise we work iteratively from expectations of current model.
- We construct an EM training algorithm, as for HMMs
- For each sentence, at each iteration, we work out expectation of how often each rule is used using inside and outside probabilities
- We assume sentences are independent and sum expectations over parses of each
- We re-estimate rules based on these ‘counts’