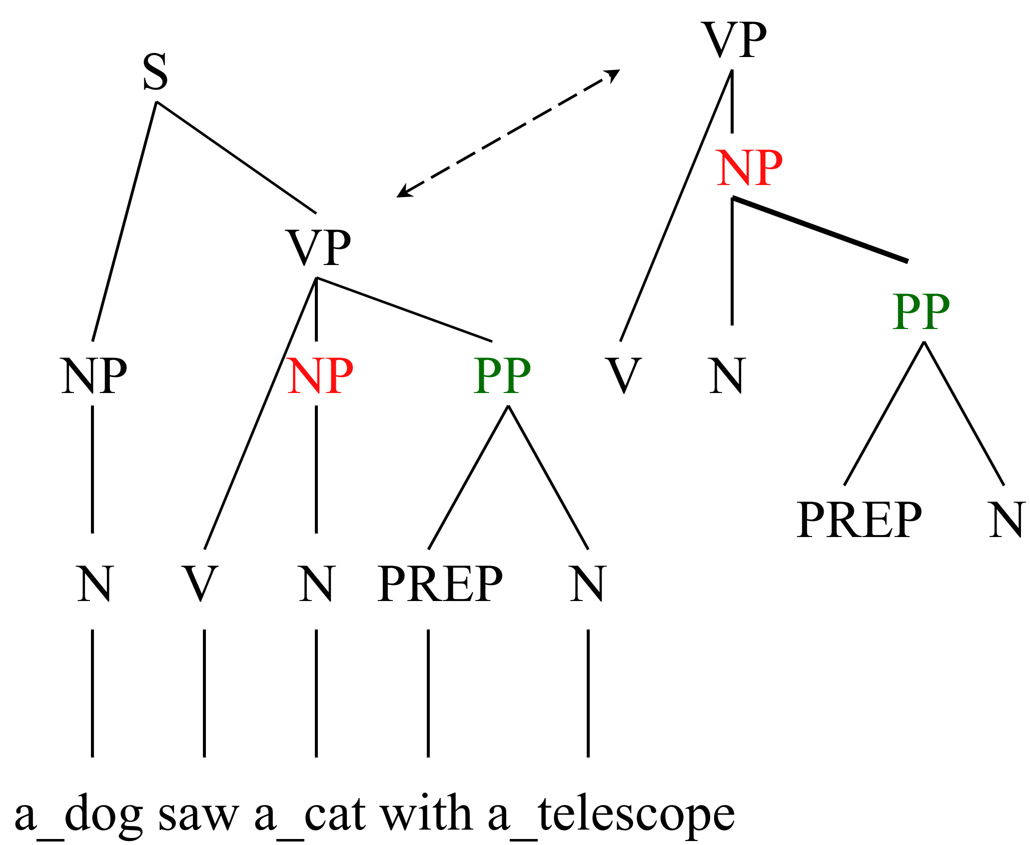
Probabilistic CFG (Introduction)

Context-free Grammars

- Chomsky hierarchy
  - Type 0 Grammars/Languages
    - rewrite rules $\alpha \rightarrow \beta$; α,β are any string of terminals and nonterminals
  - Context-sensitive Grammars/Languages
    - rewrite rules: $\alpha X\beta \rightarrow \alpha\gamma\beta$, where X is nonterminal, α,β,γ any string of terminals and nonterminals (γ must not be empty)
  - **Context-free Grammars/Lanuages**
    - rewrite rules: $X \rightarrow \gamma$, where X is nonterminal, γ any string of terminals and nonterminals
  - Regular Grammars/Languages
    - rewrite rules: $X \rightarrow \alpha\ Y$ where X,Y are nonterminals, α string of terminal symbols; Y might be missing

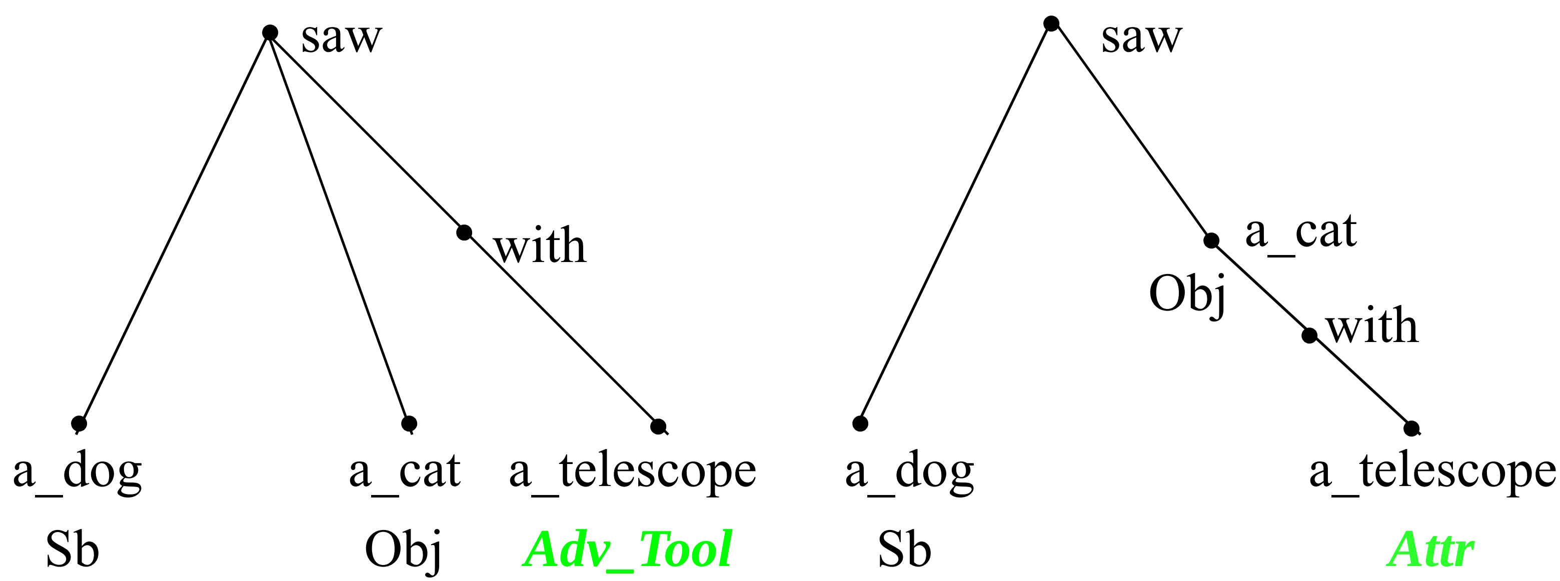
Another NLP Example

- #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



Dependency Style Example

- Same example, dependency representation



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:

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

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

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 Rule

- Rule r(i): $A \rightarrow \alpha$;
- 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:

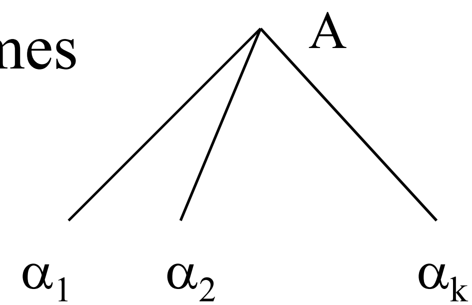
$$\sum_{r \in R_A} p(r) = 1,\ 0 \leq p(r) \leq 1$$

- Another point of view:

$$p(\alpha|A) = p(r), \text{ where } r = A \rightarrow \alpha,\ \alpha \in (N \cup T)^+$$

Estimating Probability of a Rule

- MLE from a treebank ***following a CFG grammar***
- Let's $r = A \rightarrow \alpha_1\, \alpha_2 ... \alpha_k$:
  - $p(r) = c(r) / c(A)$
  - Counting rules (c(r)): how many times



    appears in the treebank.
  - Counting nonterminals c(A):

    just count'em (in the treebank)

Using Probabilistic CFG

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:

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

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

- Probability of a string $W = (w_1, w_2, ..., w_n)$?
- Non-trivial, because there may be many trees T_j as a result of parsing W.

Probability of a String

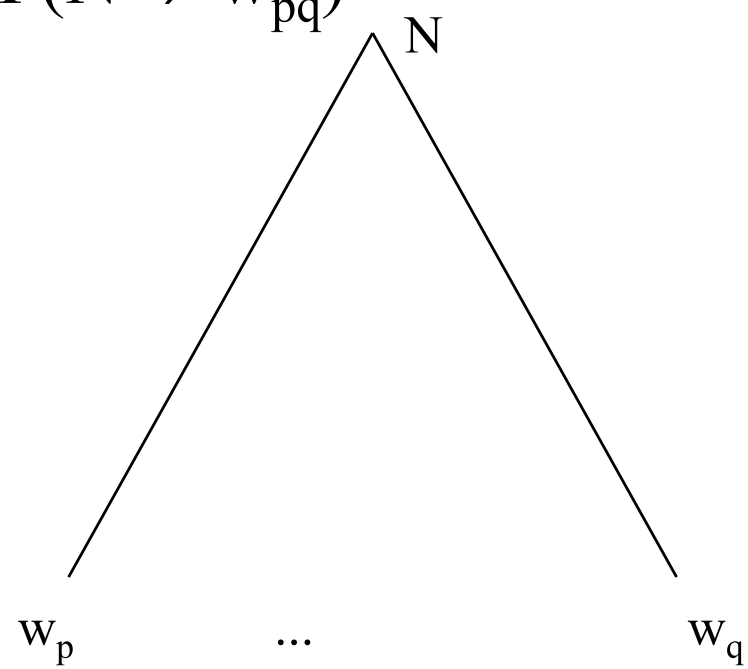
- Input string: W
- Parses: $\{T_j\}_{j=1..n}$ = Parse(W).

$$P(W) = \sum_{j=1..n} P(T_j) \quad !$$

- Impossible to use the naive method.

Inside Probability

- $\beta_N(p,q) = P(N \Rightarrow^* w_{pq})$



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

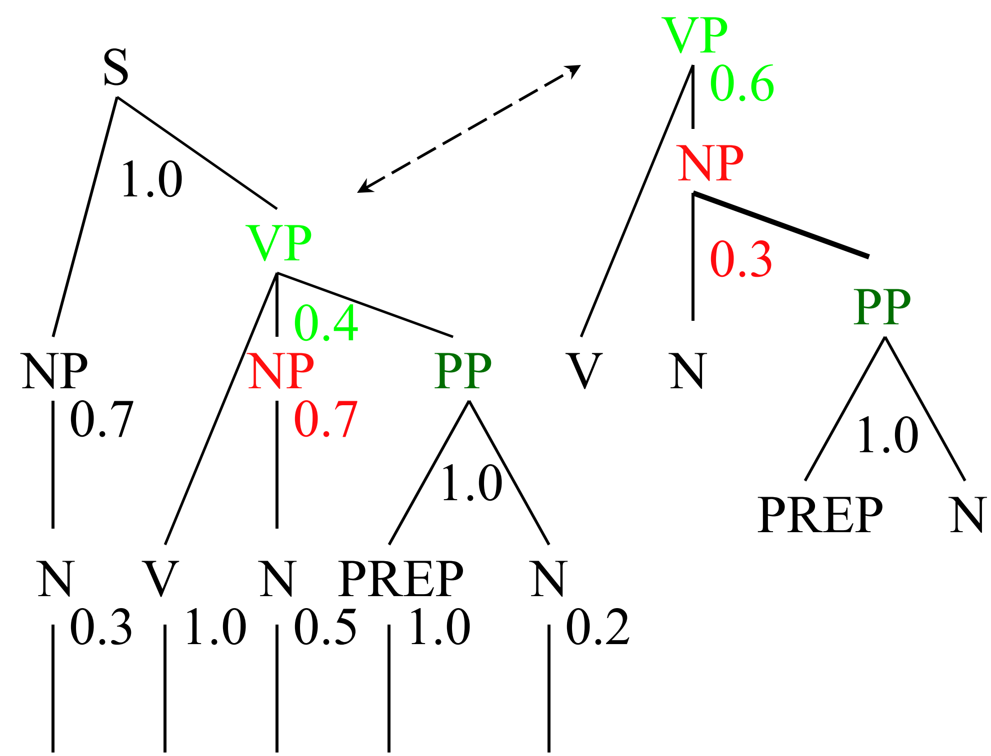
$N \rightarrow \omega$ (terminal string only)

$N \rightarrow A\ B$ (two nonterminals)

only (Chomsky Normal Form).

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 Probabilty

- 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 .0441		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 of string). Cells might have more “lines”.
- Initialize on diagonal, using $N \rightarrow \alpha$ rules.
- Recursively compute along the diagonal towards the upper right corner.