Statistical parsing 2

- Using statistical models for
  - Ambiguity resolution
    - E.g. Pp-attachment
  - Guiding parsing
  - Get the most likely parse
- Grammatical induction from corpora
- Goal: Parsing of non restricted texts with a reasonable level of accuracy (>90%) and efficiency.
- Requirements:
  - Corpora tagged (with POS): Brown, LOB, Clic-Talp
  - Corpora analyzed: Penn treebank, Susanne, Ancora

Statistical parsing 3

- Lexical approaches
  - Context free: unigram
  - Context dependent: N-gram, HMM
- Syntactic approaches
  - SCFG
- Hibrid approaches
  - Stochastic lexicalized Tags
- Computing the most likeky (most probable) parse
  - Viterbi
- Parameter learning
  - Supervised
    - Tagged corpora
  - Non supervised
    - Baum-Welch (Fw-Bw) para HMM
    - Inside-Outside para SCFG

SCFG 1

- Associate a probability to each rule
- Associate a probability to each lexical entry
- Frequent restriction CNF:
  - Binary rules $A_p \rightarrow A_q\ A_r$ matrix B_{pqr}
  - Unary rules $A_p \rightarrow b_m$ matrix U_{pm}

SCFG 2

S	⇒	NP	VP	1.0
VP	⇒	Vi		0.4
VP	⇒	Vt	NP	0.4
VP	⇒	VP	PP	0.2
NP	⇒	DT	NN	0.3
NP	⇒	NP	PP	0.7
PP	⇒	P	NP	1.0

Vi	⇒	sleeps	1.0
Vt	⇒	saw	1.0
NN	⇒	man	0.7
NN	⇒	woman	0.2
NN	⇒	telescope	0.1
DT	⇒	the	1.0
IN	⇒	with	0.5
IN	⇒	in	0.5

- Probability of a tree t with rules

$$\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2, \ldots, \alpha_n \rightarrow \beta_n$$

is

$$p(t) = \prod_{i=1}^{n} q(\alpha_i \rightarrow \beta_i)$$

where $q(\alpha \rightarrow \beta)$ is the probability for rule $\alpha \rightarrow \beta$.

SCFG 3

1. *A context-free grammar* $G = (N, \Sigma, S, R)$.

2. *A parameter*

$$q(\alpha \rightarrow \beta)$$

for each rule $\alpha \rightarrow \beta \in R$. *The parameter* $q(\alpha \rightarrow \beta)$ *can be interpreted as the conditional probabilty of choosing rule* $\alpha \rightarrow \beta$ *in a left-most derivation, given that the non-terminal being expanded is* α. *For any* $X \in N$, *we have the constraint*

$$\sum_{\alpha \rightarrow \beta \in R : \alpha = X} q(\alpha \rightarrow \beta) = 1$$

In addition we have $q(\alpha \rightarrow \beta) \geq 0$ *for any* $\alpha \rightarrow \beta \in R$.

Given a parse-tree $t \in \mathcal{T}_G$ *containing rules* $\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2, \ldots, \alpha_n \rightarrow \beta_n$, *the probability of* t *under the PCFG is*

$$p(t) = \prod_{i=1}^{n} q(\alpha_i \rightarrow \beta_i)$$

SCFG 4

- Assigns a probability to each *left-most derivation*, or parse-tree, allowed by the underlying CFG

- Say we have a sentence s, set of derivations for that sentence is $\mathcal{T}(s)$. Then a PCFG assigns a probability $p(t)$ to each member of $\mathcal{T}(s)$. i.e., *we now have a ranking in order of probability*.

- The most likely parse tree for a sentence s is

$$\arg\max_{t \in \mathcal{T}(s)} p(t)$$

SCFG 5

- **Learning.** How to obtain a PCFG from a treebank?
- **Inference.** Given a PCFG and a sentence s. Denote by $\mathcal{T}(s)$ the set of derivations that yield s.
  - How to compute the best parse for s?

$$\operatorname*{argmax}_{t \in \mathcal{T}(s)} p(t)$$

  - How to compute the probability of s?

$$\sum_{t \in \mathcal{T}(s)} p(t)$$

SCFG 6

Pros and cons of SCFG

- Some idea of the probability of a parse
  - But not very good.
- CFG cannot be learned without negative examples, SCFG can
- SCFGs provide a LM for a language
- In practice SCFG provide a worse LM than a 3-gram
  - P([N [N toy] [N [N coffee] [N grinder]]]) = P ([N [N [N cat] [N food]] [N tin]])
  - P (NP $\rightarrow$ Pro) is > in Subj position than in Obj position.

SCFG 7

- Robust
- Posibility of combining SCFG with 3-grams
- SCFG assign a lot of probability mass to short sentences (a small tree is more probable than a big one)
- Parameter estimation (probabilities)
- Problem of sparseness
- Volume

SCFG 8

- Association to the rule. Information about the point of application of the rule in the derivation tree is lost.
- Low frequency constructions are penalized
- Probability of a derivation. Contextual independence is assumed
- Posibility of relax conditional independence:
  - Sensitivity to structure
  - Lexicalization

SCFG 9

- Sensitivity to structure
  - Node expansion depends of its position in the tree

	Pronoun	Lexical
Subject	91%	9%
Object	34%	66%

  - Enrichment of a node with information of its ancestors
    - ^{S}NP is different of ^{VP}NP
  - Pronouns as arguments of ditransitive verbs
    - *I gave Charlie the book*
    - *I gave the book to Charlie*
    - *I gave you the book*
    - *? I gave the book to you*

SCFG 10

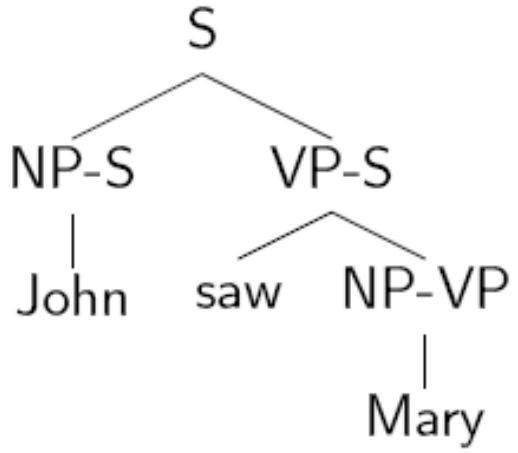
- (Head) Lexicalization
  - *put* takes both an NP and a VP
    - *Sue put* [*the book*]$_{NP}$ [*on the table*]$_{PP}$
    - * *Sue put* [*the book*]$_{NP}$
    - * *Sue put* [*on the table*]$_{PP}$
  - *like* usually takes an NP and not a PP
    - *Sue likes* [*the book*]$_{NP}$
    - * *Sue likes* [*on the table*]$_{PP}$

SCFG 11

Local Tree	*Come*	*Take*	*Think*	*Want*
VP-> V	9.5%	2.6%	4.6%	5.7%
VP-> V NP	1.1%	32.1%	0.2%	13.9%
VP-> V PP	34.5%	3.1%	7.1%	0.3%
VP- V SBAR	6.6%	0.3%	73.0%	0.2%
VP-> V S	2.2%	1.3%	4.8%	70.8%
VP->V NP S	0.1%	5.7%	0.0%	0.3%
VP->V PRT NP	0.3%	5.8%	0.0%	0.0%
VP->V PRT PP	6.1%	1.5%	0.2%	0.0%

PCFGs with Parent Annotations

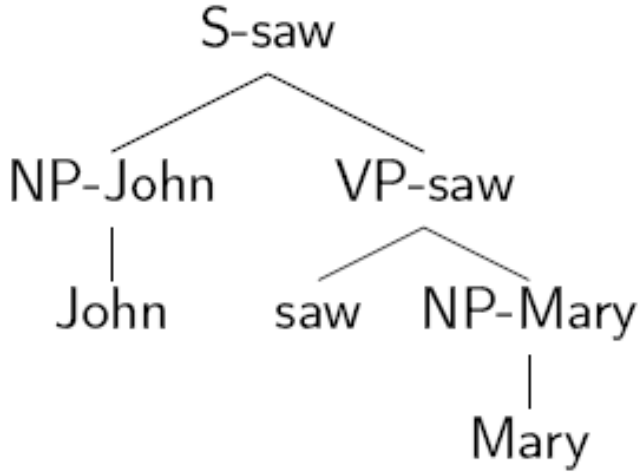
(Johnson, 1999)



$$
\begin{aligned}
P(Tree) \;=\; & q(\texttt{S -> NP-S VP-S} \mid \texttt{S}) \times \\
& q(\texttt{NP-S -> John} \mid \texttt{NP-S}) \times \\
& \ldots
\end{aligned}
$$

Lexicalized PCFGs

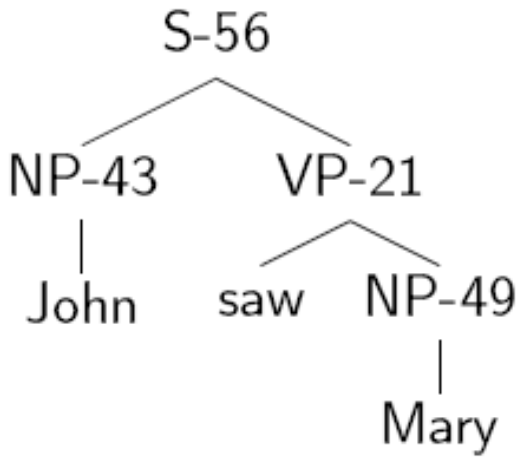
(Collins, 1999)



$$
\begin{aligned}
P(Tree) \;=\; & P(\texttt{S-saw -> NP-John VP-saw} \mid \texttt{S-saw}) \times \\
& P(\texttt{NP-John -> John} \mid \texttt{NP-John}) \times \\
& \ldots
\end{aligned}
$$

PCFGs with Latent Variables

(e.g., Petrov and Klein, 2007)



- Each non-terminals (e.g., S) is split into a number of new non-terminals (e.g., S-1, S-2, . . ., S-128)
- Latent annotations learned using EM

Parsing SCFG 1

- Two models
- **Conditional/Discriminative model :**
  - The probability of a parse tree is directly estimated

$$\hat{t} = \underset{t}{\operatorname{argmax}}\, P(t \mid s, G) \quad \text{con} \sum_{t} P(t \mid s, G) = 1$$

  - Probabilities are conditioned on a concrete sentence.
  - No sentence distribution probabilities is assumed

Parsing SCFG 2

- **Generative/joined model :**
  - Assigns probabilities to all the trees generated by the grammar. Probabilities are, thus, for language *L*:

$$\Sigma_{\{t:\text{yield}(t) \in L\}} \; P(t) = 1$$

$$\hat{t} = \underset{t}{\text{argmax}}\, P(t, s \mid G) = \underset{t}{\text{argmax}}\, P(t \mid s, G) \Big/ P(s \mid G)$$

Parsing SCFG 3

- CFG
- SCFG
  - For each rule of G, $(A \rightarrow \alpha) \in P_G$ we should be able to define a probability $P(A \rightarrow \alpha)$

$$\sum_{(A \rightarrow \alpha) \in P_G} P(A \rightarrow \alpha) = 1$$

- Probability of a tree

$$P(\psi) = \prod_{(A \rightarrow \alpha) \in P_G} P(A \rightarrow \alpha)^{f(A \rightarrow \alpha; \psi)}$$

Parsing SCFG 4

- P(t) -- Probability of a tree t (product of probabilities of the rules generating it.
- P(w_{1n}) -- Probability of a sentence is the sum of the probabilities of all the valid parse trees of the sentence

$$P(w_{1n}) = \Sigma_j P(w_{1n}, t) \text{ where } t \text{ is a parse of } w_{1n}$$
$$= \Sigma_j P(t)$$

Parsing SCFG 5

- Positional invariance:
  - The probability of a subtree is independent of its position in the derivation tree
- Context-free
- Independence from ancestors:

Parsing SCFG 6

Parameter estimation

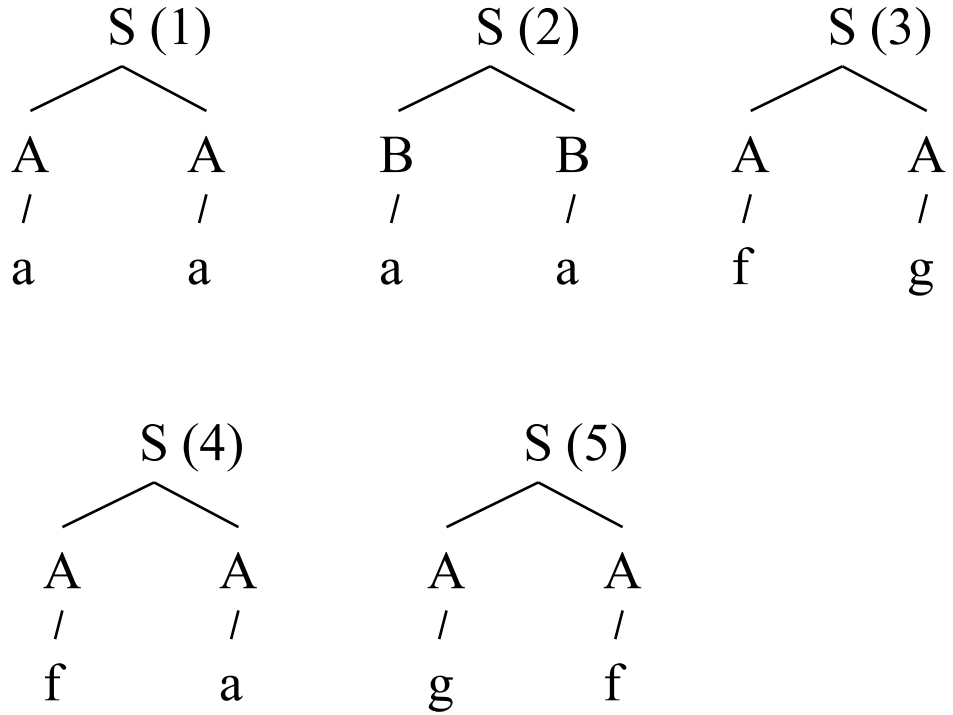
- Supervised learning
  - From a treebank
    - $\{\psi_1, \ldots, \psi_N\}$
- Non supervised learning
  - Inside/Outside (EM)
  - Similar to Baum-Welch in HMM

Treebank grammars 1

- Grammars directaly derived from a treebank
  - Charniak,1996
    - Using PTB
      - 47,000 sentences
    - Navigating PTB where each local subtree provides the left hand and right hand side of a ule
    - Precision and recall around 80%
    - around 17,500 rules
- Sekine,1997, Sekine & Grishman,1995
- Treebank grammars compactation
  - Continuous growth of the grammar size
  - Krotov et al,1999, Krotov,1998, Gaizauskas,1995

Treebank grammars 2

- Consider a treebank containing the following trees:



Treebank grammars 3

- Supose that (1) occurs 40 times, (2) occurs 10 times, (3) occurs 5 times, (4) occurs 5 times, and (5) occurs once.
- We want to induce a SCFG reflexing this treebank.
- Parameter estimation

$$\Sigma_j \, \mathrm{P}(N^i \rightarrow \zeta^j \mid N^i) = 1$$

Treebank grammars 4

- Rules

S → A A	:	40 + 5 + 5 + 1 = 51
S → B B	:	10
A → *a*	:	40 + 40 + 5 = 85
A → *f*	:	5 + 5 + 1 = 11
A → *g*	:	5 + 1 = 6
B → *a*	:	10

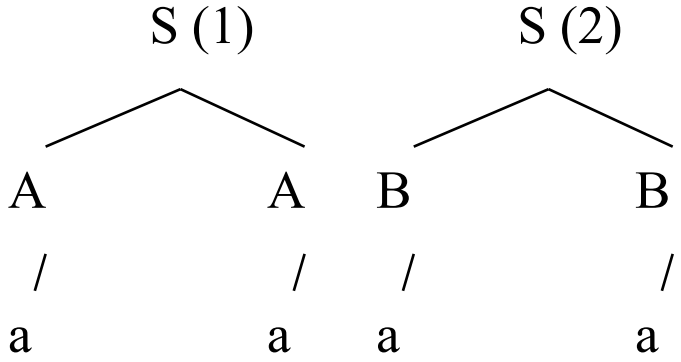
Treebank grammars 5

- Parameters maximizing global likelihood of the corpus:

G	Frequency	Total	Probability
S → A A	51	61	0.836
S → B B	10	61	0.164
A → *a*	85	102	0.833
A → *f*	11	102	0.108
A → *g*	6	102	0.059
B → *a*	10	10	1.0

Treebank grammars 6

- Given this parametrization,what is the most likely parse for "a a"?



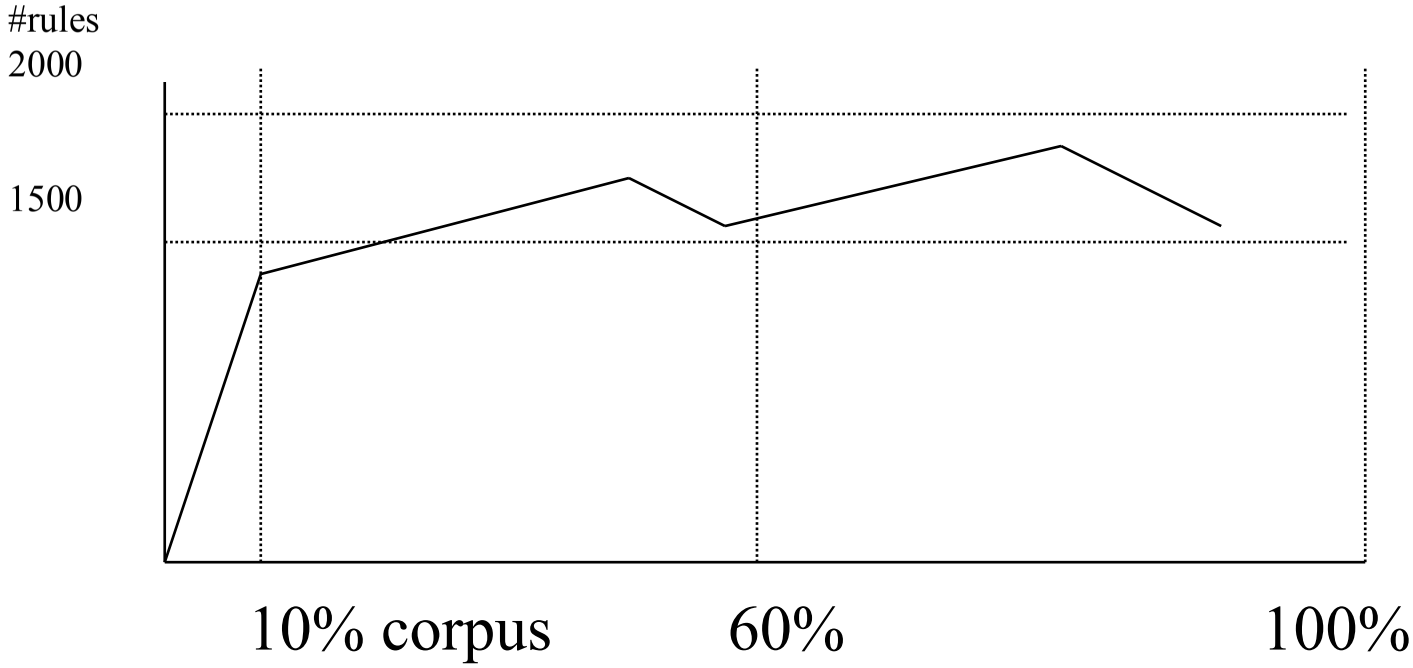
- P(1) = P(S → A A) * P(A → *a*) * P(A → *a*)
  = 0.836 * 0.833 * 0.833 = 0.580
- P(2) = P(S → B B) * P(B → *a*) * P(B → *a*)
  = 0.164 * 1.0 * 1.0 = 0.164

Treebank grammars 7

- Treebank Grammars compactation
- Partial bracketting
  - NP $\rightarrow$ DT NN CC DT NN
  - NP $\rightarrow$ NP CC NP
  - NP $\rightarrow$ DT NN
- Redundance removing (some rules can be generated from others)

Treebank grammars 8

- Applying compactation
  - 17,529 → 1,667 rules



Treebank grammars 9

- Removing non lnguistically valid rules
  - Assign probabilities (MLE) to the initial rules
  - Remove a rule except in the the probability of the structure built from its application is greater than the probability of building applicating simpler rules.
- thresholding
  - Removing rules occurring < n times

	Full	Simply thresholded	Fully compacted	Linguistically Compacted Grammar 1	Linguistically Compacted Grammar 2
Recall	70.55	70.78	30.93	71.55	70.76
precision	77.89	77.66	19.18	72.19	77.21
Grammar size	15,421	7,278	1,122	4,820	6,417

Maximum Entropy Models (ME) 1

- Maximum Entropy: alternative estimation technique.
- Able to deal with different kinds of evidence
- ME principle:
  - Do not assume anything about non-observed events.
  - Find the most uniform (maximum entropy, less informed) probability distribution that matches the observations.
- Example:

$p(a,b)$	0	1	
x	?	?	
y	?	?	
total	0.6		1.0

Observations

$p(a,b)$	0	1	
x	0.5	0.1	
y	0.1	0.3	
total	0.6		1.0

One possible $p(a,b)$

$p(a,b)$	0	1	
x	0.3	0.2	
y	0.3	0.2	
total	0.6		1.0

Max.Entropy $p(a,b)$

Maximum Entropy Models (ME) 2

- Observed facts are constraints for the desired model p.
- Constraints take the form of feature functions:

$$f_i : \varepsilon \rightarrow \{0, 1\}$$

- The desired model must satisfy the constraints:

$$E_p(f_i) = E_{\tilde{p}}(f_i) \quad \forall i$$

where:

$$E_p(f_i) = \sum_{x \in \varepsilon} p(x) f_i(x) \qquad \text{expectation of model } p.$$

$$E_{\tilde{p}}(f_i) = \sum_{x \in \varepsilon} \tilde{p}(x) f_i(x) \qquad \text{observed expectation.}$$

Maximum Entropy Models (ME) 3

- Example:

$$\varepsilon = \{x, y\} \times \{0, 1\}$$

$p(a,b)$	0	1	
x	?	?	
y	?	?	
total	0.6		1.0

- Observed fact: $p(x,0) + p(y,0) = 0.6$
- Encoded as a constraint: $E_p(f_1) = 0.6$
  where:
  - $f_1(a,b) = \begin{cases} 1 & \text{if } b = 0 \\ 0 & \text{otherwise} \end{cases}$
  - $E_p(f_1) = \sum_{(a,b) \in \{x,y\} \times \{0,1\}} p(a,b) f_1(a,b)$

Maximum Entropy Models (ME) 4

Ratnaparkhi ME parser

- ME based
- 4 procedures
  - Tag
  - Chunk
  - Build
  - Check $P_X(a \mid b)$
- where
  - X is one of the procedures
  - a is a valid action for this procedure
  - b is the context

Maximum Entropy Models (ME) 5

- A set of *templetes* is asociated to each of the procedures
- *templetas* incorpates the type of features relevant for parsing
  - *Headwords* of constituents
  - Combination of *headwords*
  - Generalizations (POS, syntactic categories)
  - Limited forms of *look-ahead*

Maximum Entropy Models (ME) 6

- Learning is very simple:
  - Simple counting rejecting features occurring < 5 times.
  - Using 40.000 sentences from *PTB* the model includes 120.000 features of type *TAG*, 230.000 of type *CHUNK*, 180.000 of type *CHECK* and 530.000 of type *BUILD* (most of them lexicalized).
- Good results:
  - recall (85.3%) precision (87.5%).

Lexicalized parsers 1

- Two types of statistics
  - Relation between the head of a phrase and the rule applyed to expand it:
    - p(r|h)
    - r = rule, h = head
  - Relation between the head of a phrase and another sub-component:
    - p(h|m,t)
    - h = head of the sub-component, m = head of the component, t = type of the sub-component
  - probability of a lexicalized parse tree:
    - $p(s,\pi) = p(h(c)|m(c),t) \cdot p(r(c)|h(c))$

Lexicalized parsers 2

- Other useful statistics
  - The type of the parent conditions the probability of the rule (Charniak, 1997)
  - Immediate left context (Collins,1996)
  - Classification of NP in a VP as optional or mandatory (Collins,1997)
  - ...

Lexicalized parsers 3

- parser based in SCFG.

Collins parser

$$\psi_{OPT} = \underset{\psi}{\text{argmax}}\ P(\psi \mid s) = \underset{\psi}{\text{argmax}}\ P(\psi, s)$$

- where ψ_{OPT} is the most probable parser and ψ represents whatever parse tree of sentence *s*. If ψ is obtained applying *n* rules of type $A \rightarrow \alpha$ for producing *s*, we get:

$$P(\psi, s) = \prod_{i=1}^{n} P((A \rightarrow \alpha)_i) = \prod_{i=1}^{n} P(\alpha_i \mid A_i)$$

Lexicalized parsers 4

- The rules of the grammar are of the form that for each rule the left side has attached a word (*head*) taken from one of the components of the right side.
- *P* parent component, *H* head, L_i and R_j components to the left (left modifiers) and right (right modifiers) of the head, *h*, *head*, l_i and r_j words associated to these components:
  - $P(h) \rightarrow L_n(l_n) \ldots L_1(l_1)\ H(h)\ R_1(r_1) \ldots R_m(r_m)$

Lexicalized parsers 5

- The initial model, lexicalized, cannot be scaled up and Collins decompose it into 3 simpler models which can be learned from the *PTB* and can generate the righ side part
- The first generates *H* with probability P(H|P,h) given a category *P* and a headword *h*.

Lexicalized parsers 5

- Then the right modifiers at distance i = 1, …, m+1 (where $R_{m+1}(r_{m+1}) = \text{STOP}$):
-

$$\prod_{i=1}^{m+1} P_r(R_i(r_i) \mid P, h, H)$$

- Now conditioned not only to *P* and *h* but also to *H*
- Finally the left modifiers:

$$\prod_{i=1}^{n+1} P_l(L_i(l_i) \mid P, h, H)$$

Lexicalized parsers 6

- For instance, the probability associated to the following rule:
  - S(bought) $\rightarrow$ NP(week) NP(Mark) VP(bought)
- would be:

$$P_h(\text{VP} \mid \text{S}, \text{bought}) \cdot P_l(\text{NP(Mark)} \mid \text{S}, \text{bought}, \text{VP}) \cdot$$
$$P_l(\text{NP(week)} \mid \text{S}, \text{bought}, \text{VP}) \cdot P_l(\text{STOP} \mid \text{S}, \text{bought}, \text{VP}) \cdot$$
$$P_h(\text{STOP} \mid \text{S}, \text{bought}, \text{VP}) \cdot$$

Lexicalized parsers 7

- The parser was learned with 40.000 sentences of PTB
- Results:
  - 87.4% recall and 88.1% precision.
- In Collins (1997) the model is improved with linguistic information on the distinction between complements and adjunts (model 2) and constituent moving, relative sentences (modelo 3).
- With these improvements the parser reaches 88.1% of recall and 88.6% of precision.