

Probabilistic context-free parsing

Parsing
ISCL-BA-06

Çağrı Çöltekin
ccoltekin@sfs.uni-tuebingen.de

University of Tübingen
Seminar für Sprachwissenschaft

Winter Semester 2020/21

Context-free grammars

recap

- Context free (CF) grammars are most practically useful grammars in the Chomsky hierarchy
- Most of the parsing theory (and practice) is build on parsing CF languages
- The context-free rules have the form

$$A \rightarrow \alpha$$

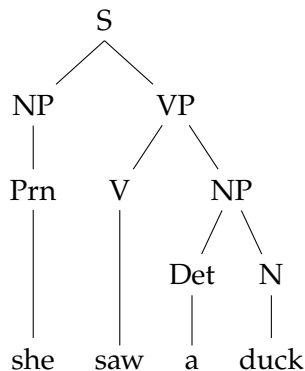
where A is a single non-terminal symbol and α is a (possibly empty) sequence of terminal or non-terminal symbols

An example context-free grammar

$S \rightarrow NP VP$
 $S \rightarrow Aux NP VP$
 $NP \rightarrow Det N$
 $NP \rightarrow Prn$
 $NP \rightarrow NP PP$
 $VP \rightarrow V NP$
 $VP \rightarrow V$
 $VP \rightarrow VP PP$
 $PP \rightarrow Prp NP$
 $N \rightarrow duck$
 $N \rightarrow park$
 $N \rightarrow parks$
 $V \rightarrow duck$
 $V \rightarrow ducks$
 $V \rightarrow saw$
 $Prn \rightarrow she | her$
 $Prp \rightarrow in | with$
 $Det \rightarrow a | the$

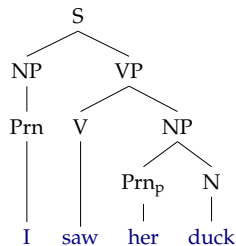
Derivation of sentence 'she saw a duck'

$S \Rightarrow NP VP$
 $NP \Rightarrow Prn$
 $Prn \Rightarrow she$
 $VP \Rightarrow V NP$
 $V \Rightarrow saw$
 $NP \Rightarrow Det N$
 $Det \Rightarrow a$
 $N \Rightarrow duck$



Representations of a context-free parse tree

A parse tree:



A history of derivations:

- $S \Rightarrow NP VP$
- $NP \Rightarrow Prn$
- $Prn \Rightarrow I$
- $VP \Rightarrow V NP$
- $V \Rightarrow \text{saw}$
- $NP \Rightarrow Prn_p N$
- $Prn_p \Rightarrow \text{her}$
- $N \Rightarrow \text{duck}$

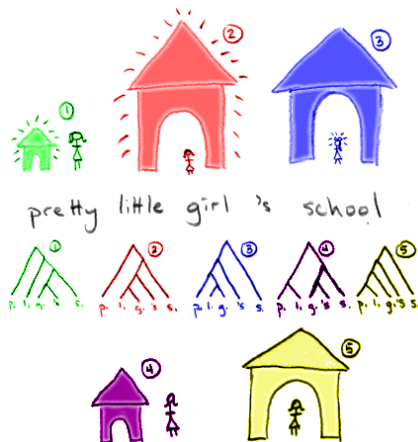
A sequence with (labeled) brackets

$$\left[{}_S \left[{}_{NP} \left[{}_{Prn} I \right] \right] \left[{}_{VP} \left[{}_V \text{saw} \right] \left[{}_{NP} \left[{}_{Prn_p} \text{her} \right] \left[{}_N \text{duck} \right] \right] \right] \right]$$

Parsing with context-free grammars

- Parsing can be
 - top down: start from S , search for derivation that leads to the input
 - bottom up: start from input, try to *reduce* it to S
- Naive search for both recognition/parse is intractable
- Dynamic programming methods allow polynomial time *recognition*
 - CKY bottom-up, requires Chomsky normal form
- Earely top-down (with bottom-up filtering), works with unrestricted grammars
 - $O(n^3)$ time complexity (for recognition)
- Chart parsers are (reasonably) efficient, and they can represent ambiguity in their output
- However, they do not help with *resolving ambiguity*

Natural languages are ambiguous



Some types of ambiguities

- Lexical ambiguity
 - She is looking for a match
 - We saw her duck
- Attachment ambiguity
 - I saw the man with a telescope
 - Panda eats bamboo shoots and leaves
- Local ambiguity (garden path sentences)
 - The horse raced past the barn fell
 - The old man the boats
 - Fat people eat accumulates

Ambiguity and the parsers

- Given a grammar, chart parsers (e.g., CKY, Early) can parse natural language sentences relatively efficiently
- These parsers also represent all possible parse trees in their chart/output efficiently
- However, they have nothing to say about which of these parses are the most likely one.
- The task of selecting the best parse among many is called disambiguation
- In almost all practical uses, parsers are combined with disambiguators

We do not recognize many ambiguities

- Time flies like an arrow
- Outside of a dog, a book is a man's best friend
- One morning I shot an elephant in my pajamas
- Don't eat the pizza with a knife and fork

A parser, nevertheless, produces multiple parses for these sentences.

We do not recognize many ambiguities

- Time flies like an arrow; fruit flies like a banana
- Outside of a dog, a book is a man's best friend
- One morning I shot an elephant in my pajamas
- Don't eat the pizza with a knife and fork

A parser, nevertheless, produces multiple parses for these sentences.

We do not recognize many ambiguities

- Time flies like an arrow; fruit flies like a banana
- Outside of a dog, a book is a man's best friend; inside it's too hard to read
- One morning I shot an elephant in my pajamas
- Don't eat the pizza with a knife and fork

A parser, nevertheless, produces multiple parses for these sentences.

We do not recognize many ambiguities

- Time flies like an arrow; fruit flies like a banana
- Outside of a dog, a book is a man's best friend; inside it's too hard to read
- One morning I shot an elephant in my pajamas. How he got in my pajamas, I don't know.
- Don't eat the pizza with a knife and fork

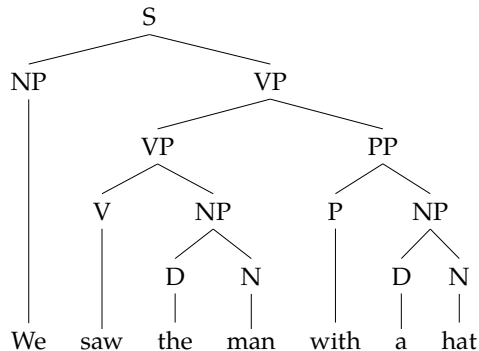
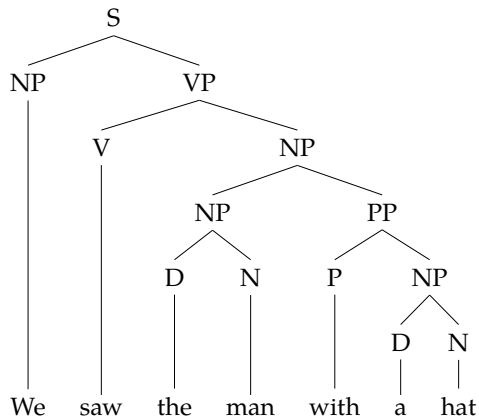
A parser, nevertheless, produces multiple parses for these sentences.

We do not recognize many ambiguities

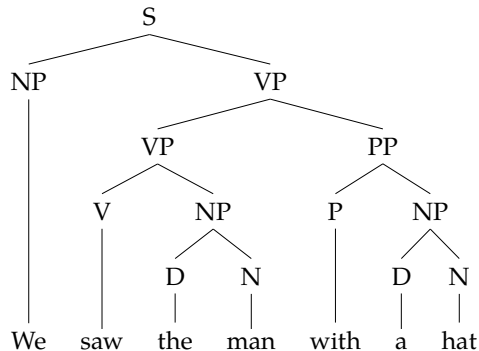
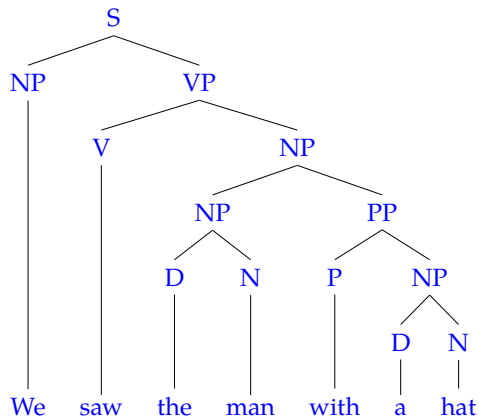
- Time flies like an arrow; fruit flies like a banana
- Outside of a dog, a book is a man's best friend; inside it's too hard to read
- One morning I shot an elephant in my pajamas. How he got in my pajamas, I don't know.
- Don't eat the pizza with a knife and fork; the one with mushrooms is better.

A parser, nevertheless, produces multiple parses for these sentences.

The task: choosing the most plausible parse



The task: choosing the most plausible parse



Statistical parsing

- Find the most plausible parse of an input string given all possible parses
- We need a scoring function, for each parse, given the input
- We typically use probabilities for scoring, task becomes finding the parse (or tree), t , given the input string \mathbf{w}

$$t_{\text{best}} = \arg \max_t P(t \mid \mathbf{w})$$

- Note that some ambiguities need a larger context than the sentence to be resolved correctly

Probability refresher (1)

- Probability is a measure of (un)certainty of an event
- We quantify the probability of an event with a number between 0 and 1
 - 0 the event is impossible
 - 0.5 the event is as likely to happen (or happened) as it is not
 - 1 the event is certain
- All possible outcomes of a trial (experiment or observation) is called the *sample space* (Ω)

Axioms of probability states that

1. $P(E) \in \mathbb{R}, P(E) \geq 0$
2. $P(\Omega) = 1$
3. For *disjoint* events E_1 and E_2 , $P(E_1 \cup E_2) = P(E_1) + P(E_2)$

Probability refresher (2)

Joint and conditional probabilities, chain rule

- Joint probability of two events is noted as $P(x, y)$
- The conditional probability is defined as

$$P(x|y) = \frac{P(x,y)}{P(y)} \text{ or } P(x, y) = P(x|y)P(y)$$

- If the events x and y are independent,

$$P(x|y) = P(x), P(y|x) = p(y), P(x, y) = P(x)P(y)$$

- For more than two variables (chain rule):

$$P(x, y, z) = P(z|x, y)P(y|x)P(x) = P(x|y, z)P(y|z)P(z) = \dots$$

- If all are independent

$$P(x, y, z) = P(x)P(y)P(z)$$

Probabilistic context free grammars (PCFG)

- A probabilistic context free grammar augments a CFG by adding a probability value to each rule

$$A \rightarrow \alpha \quad [p]$$

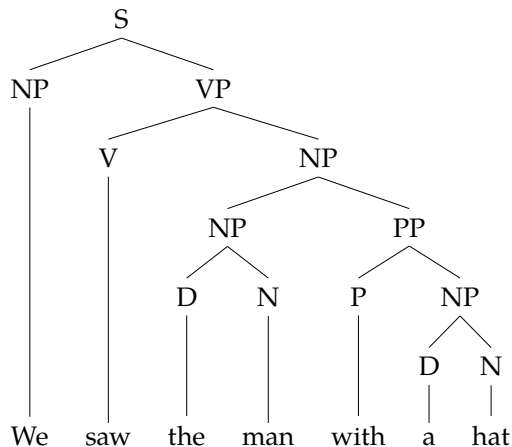
where A is a non-terminal, α is string of terminals and non-terminals, and p is the probability associated with the rule

- Like CFGs, a PCFG accepts a sentence if it can be derived from S with rules $R_1 \dots R_k$
- The probability of a parse tree t of input string \mathbf{w} , $P(t \mid \mathbf{w})$, corresponding to the derivation $R_1 \dots R_k$ is

$$P(t \mid \mathbf{w}) = \prod_1^k p(R_i)$$

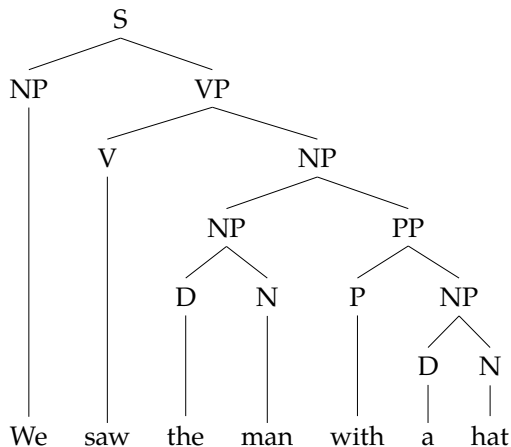
where $p(R_i)$ is the probability of the rule R_i

PCFG example (1)



S	→ NP VP	1.0
NP	→ D N	0.7
NP	→ NP PP	0.2
NP	→ We	0.1
VP	→ V NP	0.9
VP	→ VP PP	0.1
PP	→ P NP	1.0
N	→ hat	0.2
N	→ man	0.8
V	→ saw	1.0
P	→ with	1.0
D	→ a	0.6
D	→ the	0.4

PCFG example (1)

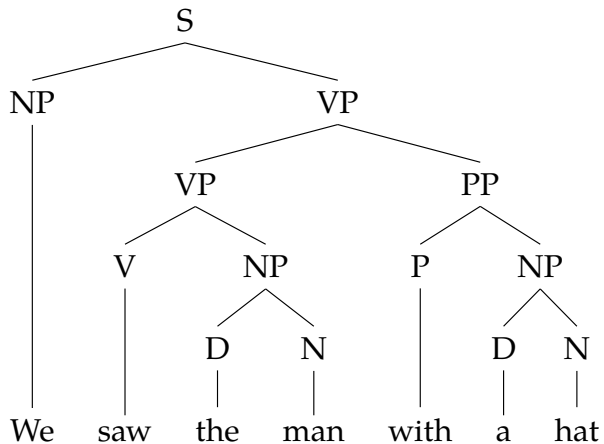


$$P(t) = 1.0 \times 0.1 \times 0.9 \times 1.0 \times 0.2 \times 0.7 \times 0.4 \times 0.8 \times 1.0 \times 1.0 \times 0.7 \times 0.6 \times 0.2$$

$$= 0.000263424$$

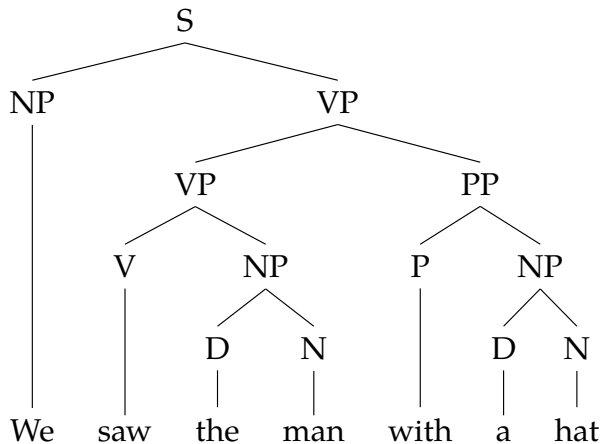
S	→ NP VP	1.0
NP	→ D N	0.7
NP	→ NP PP	0.2
NP	→ We	0.1
VP	→ V NP	0.9
VP	→ VP PP	0.1
PP	→ P NP	1.0
N	→ hat	0.2
N	→ man	0.8
V	→ saw	1.0
P	→ with	1.0
D	→ a	0.6
D	→ the	0.4

PCFG example (2)



S	→ NP VP	1.0
NP	→ D N	0.7
NP	→ NP PP	0.2
NP	→ We	0.1
VP	→ V NP	0.9
VP	→ VP PP	0.1
PP	→ P NP	1.0
N	→ hat	0.2
N	→ man	0.8
V	→ saw	1.0
P	→ with	1.0
D	→ a	0.6
D	→ the	0.4

PCFG example (2)



$$\begin{aligned}
 P(t) &= 1.0 \times 0.1 \times \mathbf{0.1} \times 0.9 \times 1.0 \times 0.7 \times 0.4 \times 0.8 \times 1.0 \times 1.0 \times 0.7 \times 0.6 \times 0.2 \\
 &= 0.0001693440
 \end{aligned}$$

S	→ NP VP	1.0
NP	→ D N	0.7
NP	→ NP PP	0.2
NP	→ We	0.1
VP	→ V NP	0.9
VP	→ VP PP	0.1
PP	→ P NP	1.0
N	→ hat	0.2
N	→ man	0.8
V	→ saw	1.0
P	→ with	1.0
D	→ a	0.6
D	→ the	0.4

Where do the rule probabilities come from?

- Supervised: estimate from a treebank, e.g., using maximum likelihood estimation
- Unsupervised: expectation-maximization (EM)

PCFGs - an interim summary

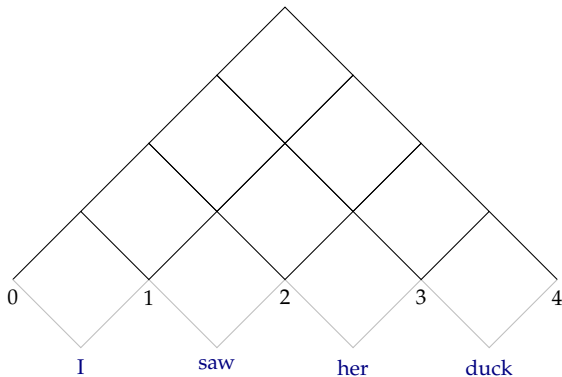
- PCFGs assign probabilities to parses based on CFG rules used during the parse
- PCFGs assume that the rules are independent
- PCFGs are generative models, they assign probabilities to $P(t, \mathbf{w})$, we can calculate the probability of a sentence by

$$P(\mathbf{w}) = \sum_t P(t, \mathbf{w}) = \sum_t P(t)$$

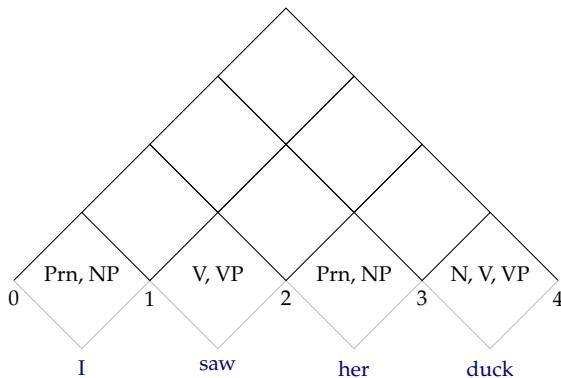
PCFG chart parsing

- Both CKY and Earley algorithms can be adapted to PCFG parsing
- CKY matches PCFG parsing quite well
 - to get the best parse, store the constituent with the highest probability in every cell of the chart
 - to get n-best best parse (beam search), store the n-best constituents in every cell in the chart

CKY for PCFG parsing



CKY for PCFG parsing



$$P(\text{Prn}_{01}) = P(\text{Prn} \rightarrow \text{I})$$

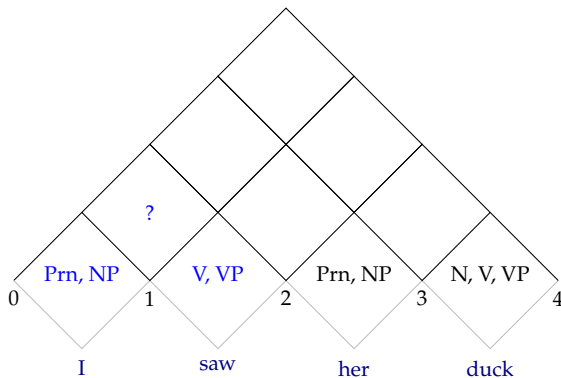
$$P(\text{V}_{12}) = P(\text{V} \rightarrow \text{saw})$$

$$P(\text{NP}_{01}) = P(\text{NP} \rightarrow \text{I})$$

$$P(\text{VP}_{12}) = P(\text{VP} \rightarrow \text{saw})$$

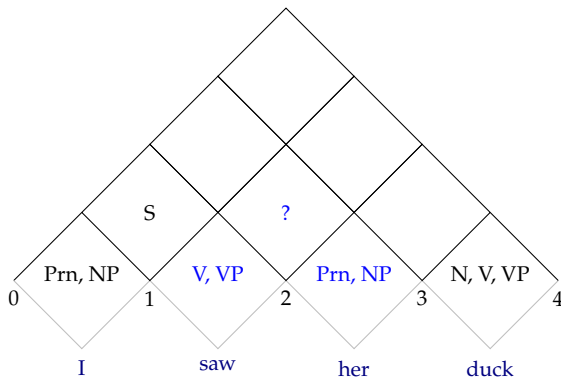
...

CKY for PCFG parsing

 $S \rightarrow NP VP$


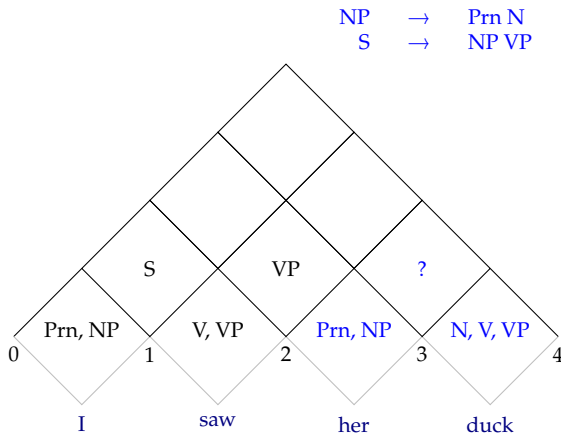
$$P(S_{02} \Rightarrow NP_{01} VP_{12}) = P(NP_{01})P(VP_{12})P(S \rightarrow NP VP)$$

CKY for PCFG parsing

$$\text{VP} \rightarrow \text{V NP}$$


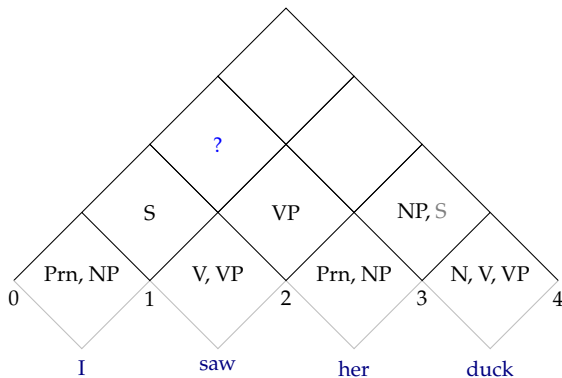
$$P(\text{VP}_{13} \Rightarrow \text{V}_{12}\text{NP}_{23}) = P(\text{V}_{12})P(\text{NP}_{23})P(\text{VP} \rightarrow \text{V NP})$$

CKY for PCFG parsing

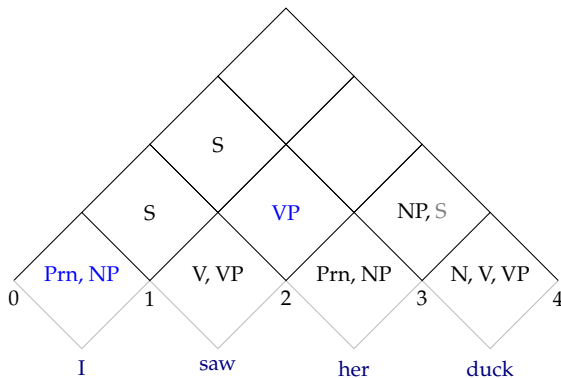


$$\begin{aligned}
 P(\text{NP}_{24} \Rightarrow \text{Prn}_{23} \text{N}_{34}) &= P(\text{Prn}_{23}) P(\text{N}_{34}) P(\text{Prn} \rightarrow \text{Prn N}) \\
 &> \\
 P(\text{S}_{24} \Rightarrow \text{NP}_{23} \text{VP}_{34}) &= P(\text{NP}_{23}) P(\text{VP}_{34}) P(\text{S} \rightarrow \text{NP VP})
 \end{aligned}$$

CKY for PCFG parsing

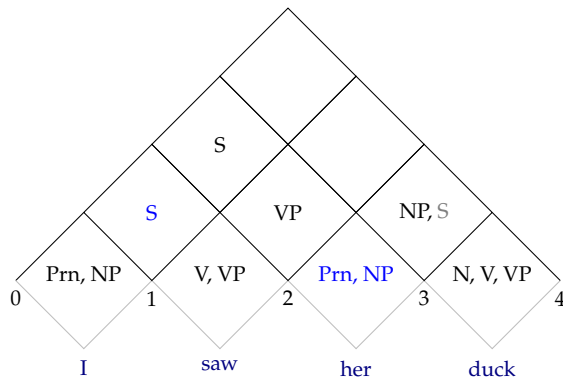


CKY for PCFG parsing

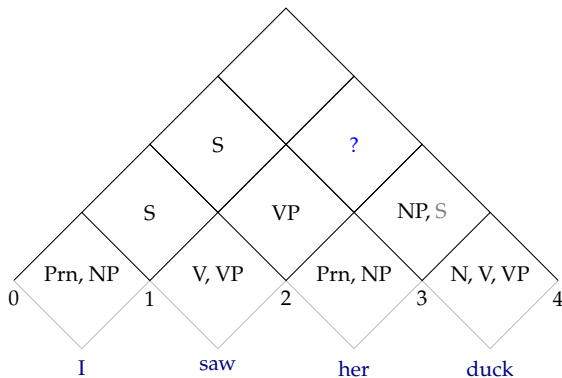
 $S \rightarrow NP VP$


$$P(S_{03} \Rightarrow NP_{01} VP_{23}) = P(NP_{01})P(VP_{13})P(S \rightarrow NP VP)$$

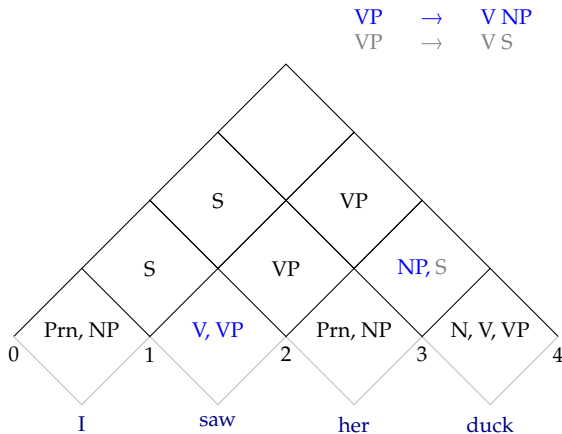
CKY for PCFG parsing



CKY for PCFG parsing

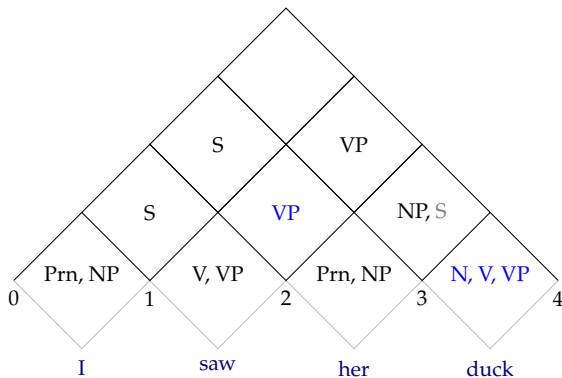


CKY for PCFG parsing

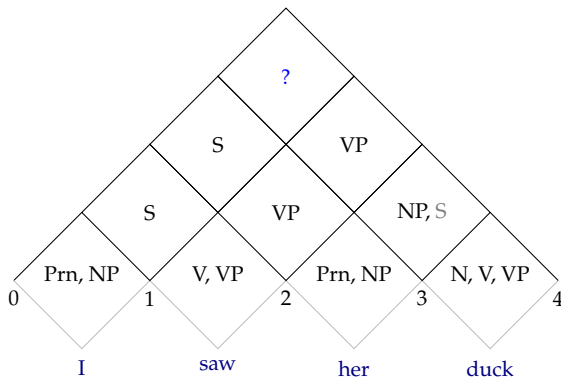


$$P(\text{VP}_{14} \Rightarrow \text{V}_{12}\text{NP}_{24}) = P(\text{V}_{12})P(\text{NP}_{24})P(\text{VP} \rightarrow \text{V NP})$$

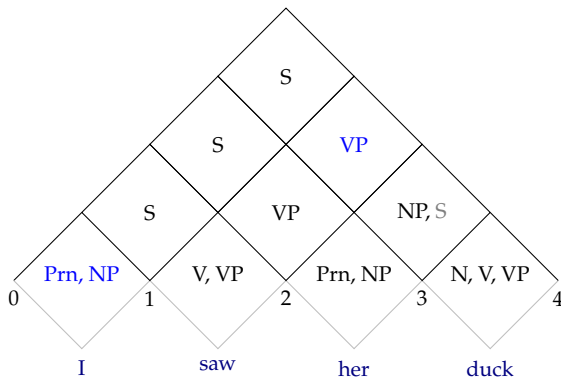
CKY for PCFG parsing



CKY for PCFG parsing

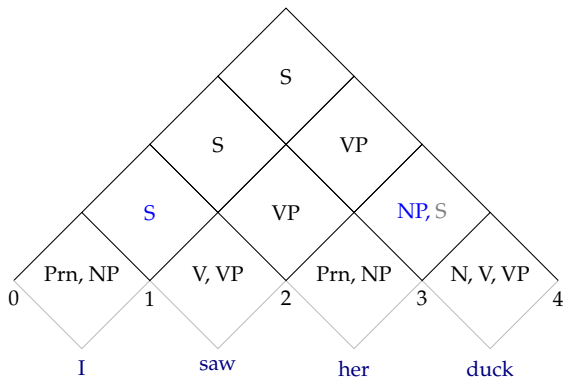


CKY for PCFG parsing

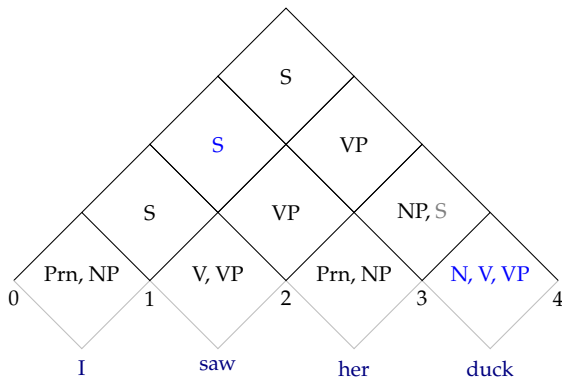
 $S \rightarrow NP VP$


$$P(S_{14} \Rightarrow NP_{01} VP_{14}) = P(NP_{01})P(VP_{14})P(S \rightarrow NP VP)$$

CKY for PCFG parsing



CKY for PCFG parsing



What makes the difference in PCFG probabilities?

S	⇒ NP VP	1.0
NP	⇒ We	0.1
VP	⇒ VP PP	0.1
VP	⇒ V NP	0.8
V	⇒ saw	1.0
NP	⇒ D N	0.7
D	⇒ the	0.4
N	⇒ man	0.8
PP	⇒ P NP	1.0
P	⇒ with	1.0
NP	⇒ D N	0.7
D	⇒ a	0.6
N	⇒ hat	0.2

S	⇒ NP VP	1.0
NP	⇒ We	0.1
VP	⇒ V NP	0.7
V	⇒ saw	1.0
NP	⇒ NP PP	0.2
NP	⇒ D N	0.7
D	⇒ the	0.4
N	⇒ man	0.8
PP	⇒ P NP	1.0
P	⇒ with	1.0
NP	⇒ D N	0.7
D	⇒ a	0.6
N	⇒ hat	0.2

What makes the difference in PCFG probabilities?

S	⇒ NP VP	1.0
NP	⇒ We	0.1
VP	⇒ VP PP	0.1
VP	⇒ V NP	0.8
V	⇒ saw	1.0
NP	⇒ D N	0.7
D	⇒ the	0.4
N	⇒ man	0.8
PP	⇒ P NP	1.0
P	⇒ with	1.0
NP	⇒ D N	0.7
D	⇒ a	0.6
N	⇒ hat	0.2

S	⇒ NP VP	1.0
NP	⇒ We	0.1
VP	⇒ V NP	0.7
V	⇒ saw	1.0
NP	⇒ NP PP	0.2
NP	⇒ D N	0.7
D	⇒ the	0.4
N	⇒ man	0.8
PP	⇒ P NP	1.0
P	⇒ with	1.0
NP	⇒ D N	0.7
D	⇒ a	0.6
N	⇒ hat	0.2

The parser's choice would not be affected by lexical items!

What is wrong with PCFGs?

- In general: the assumption of independence
- The parents affect the correct choice for children, for example, in English NP \rightarrow Prn is more likely in the subject position
- The lexical units affect the correct decision, for example:
 - We eat the pizza with hands
 - We eat the pizza with mushrooms
- Additionally: PCFGs use local context, difficult to incorporate arbitrary/global features for disambiguation

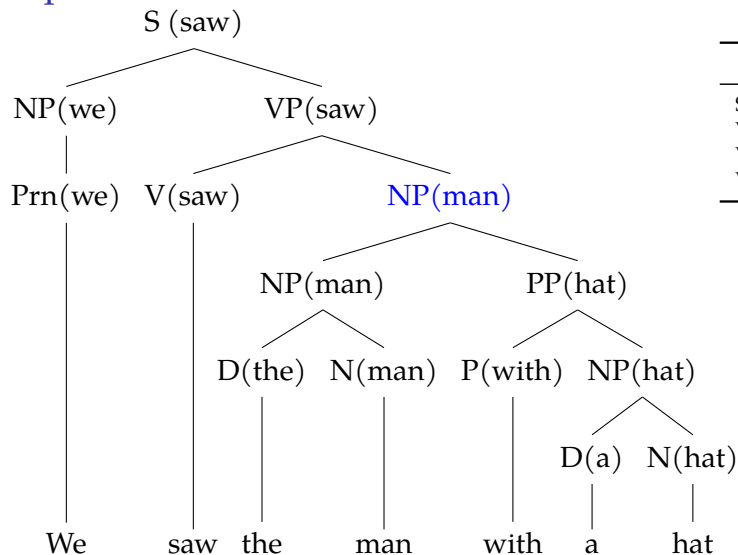
Solutions to PCFG problems

- Independence assumptions can be relaxed by either
 - Parent annotation
 - Lexicalization
 - Reranking
- To condition on arbitrary/global information: discriminative models
- Most practical PCFG parsers are lexicalized, and often use a re-ranker conditioning on other (global) features

Lexicalizing PCFGs

- Replace non-terminal X with $X(h)$, where h is a tuple with the lexical word and its POS tag
- Now the grammar can capture (head-driven) lexical dependencies
- But number of nonterminals grow by $|V| \times |T|$
- Estimation becomes difficult (many rules, data sparsity)
- Some treebanks (e.g., Penn Treebank) do not annotate heads, they are automatically annotated (based on heuristics)

Example lexicalized derivation



Example rules:

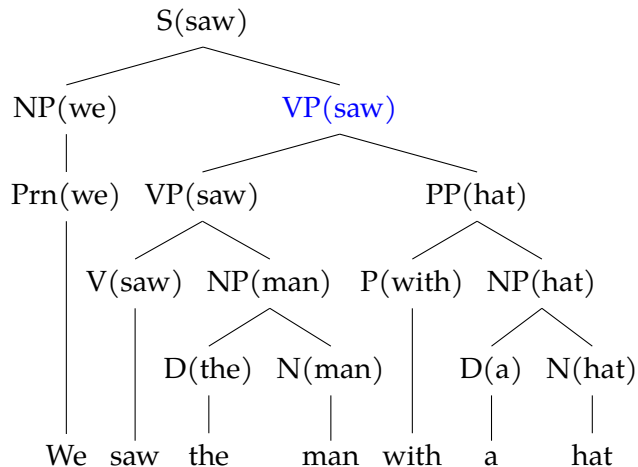
$S(\text{saw}) \rightarrow NP(\text{we}) VP(\text{saw})$

$VP(\text{saw}) \rightarrow V(\text{saw}) NP(\text{man})$

$VP(\text{saw}) \rightarrow VP(\text{saw}) PP(\text{hat})$

$VP(\text{saw}) \rightarrow VP(\text{saw}) PP(\text{telescope})$

Example lexicalized derivation



Example rules:

$S(\text{saw}) \rightarrow NP(\text{we}) VP(\text{saw})$

$VP(\text{saw}) \rightarrow V(\text{saw}) NP(\text{man})$

$VP(\text{saw}) \rightarrow VP(\text{saw}) PP(\text{hat})$

$VP(\text{saw}) \rightarrow VP(\text{saw}) PP(\text{telescope})$

Evaluating the parser output

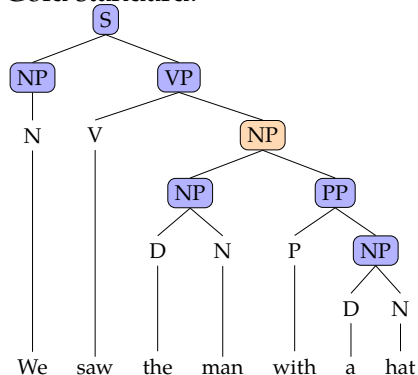
- A parser can be evaluated
 - extrinsically based on its effect on a task (e.g., machine translation) where it is used
 - intrinsically based on the match with ideal parsing
- The typically evaluation (intrinsic) is based on a *gold standard* (GS)
- Exact match is often
 - very difficult to achieve (think about a 50-word newspaper sentence)
 - not strictly necessary (recovering parts of the parse can be useful for many purposes)

Parser evaluation metrics

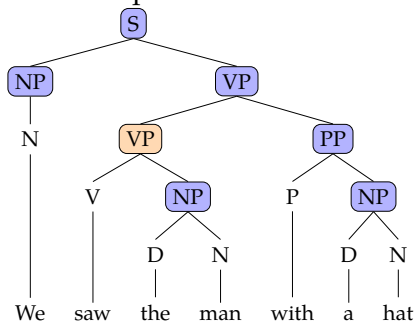
- Common evaluation metrics are (PARSEVAL):
 - precision the ratio of correctly predicted nodes
 - recall the nodes (in GS) that are predicted correctly
 - f-measure harmonic mean of precision and recall $\left(\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \right)$
- The measures can be
 - unlabeled the spans of the nodes are expected to match
 - labeled the node label should also match
- Crossing brackets (or average non-crossing brackets)
 - (We (saw (them (with binoculars)))
 - (We ((saw them) (with binoculars)))
- Measures can be averaged per constituent (micro average), or over sentences (macro average)

PARSEVAL example

Gold standard:



Parser output:



$$\text{precision} = \frac{6}{7} \quad \text{recall} = \frac{6}{7} \quad \text{f-measure} = \frac{6}{7}$$

Problems with PARSEVAL metrics

- PARSEVAL metrics favor certain type of structures
 - Results are surprisingly well for flat tree structures (e.g., Penn treebank)
 - Results of some mistakes are catastrophic (e.g., low attachment)
- Not all mistakes are equally important for semantic distinctions
- Some alternatives:
 - Extrinsic evaluation
 - Evaluation based on extracted dependencies

Summary

- PCFG is a simple attempt to augment CFG with probabilities
- PCFG parsing alone is suboptimal: independence assumptions are too strong
- Solutions include (a combination of) lexicalization, parent annotation and re-ranking
- Reading suggestion: Jurafsky and Martin (2009, Chapter 14)

Summary

- PCFG is a simple attempt to augment CFG with probabilities
- PCFG parsing alone is suboptimal: independence assumptions are too strong
- Solutions include (a combination of) lexicalization, parent annotation and re-ranking
- Reading suggestion: Jurafsky and Martin (2009, Chapter 14)

Next:

- Dependency grammars and dependency parsing

Acknowledgments, references, additional reading material



Grune, Dick and Ceriel J.H. Jacobs (2007). *Parsing Techniques: A Practical Guide*. second. Monographs in Computer Science. The first edition is available at http://dickgrune.com/Books/PTAPG_1st_Edition/BookBody.pdf. Springer New York. ISBN: 9780387689548.



Jurafsky, Daniel and James H. Martin (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. second. Pearson Prentice Hall. ISBN: 978-0-13-504196-3. URL: <http://web.stanford.edu/~jurafsky/slp3/>.

