# Foundations for Natural Language Processing
# Lecture 16
# Syntax and Parsing: Constituent Parsing (part 3)

Ivan Titov
(with slides from Ivan Titov, Dan Klein and Marco Kuhlmann)

School of informatics

# Last week

- Phrase-structure (aka constituent) trees
- (Probabilistic) Context free grammars
- CKY algorithm for CFGs

- Today:
  - CKY for PCFGs
  - Evaluation
  - Beyond ”Vanilla” treebank PCFGs

# Recap: PCFGs

Associate probabilities with the rules $p(X \rightarrow \alpha)$:

$$\forall\ X \rightarrow \alpha \in R: \quad 0 \leq p(X \rightarrow \alpha) \leq 1$$

$$\forall X \in N: \quad \sum_{\alpha: X \rightarrow \alpha \in R} p(X \rightarrow \alpha) = 1$$

Now we can score a tree as a product of probabilities corresponding to the used rules

| | | |
|---|---|---|
| $S \rightarrow NP\ \ VP$ | 1.0 | (NP A girl) (VP ate a sandwich) |
| $VP \rightarrow V$ | 0.2 | |
| $VP \rightarrow V\ \ NP$ | 0.4 | (VP ate) (NP a sandwich) |
| $VP \rightarrow VP\ \ PP$ | 0.4 | (VP saw a girl) (PP with …) |
| $NP \rightarrow NP\ \ PP$ | 0.3 | (NP a girl) (PP with ….) |
| $NP \rightarrow D\ \ N$ | 0.5 | (D a) (N sandwich) |
| $NP \rightarrow PN$ | 0.2 | |
| $PP \rightarrow P\ \ NP$ | 1.0 | (P with) (NP with a sandwich) |

| | |
|---|---|
| $N \rightarrow girl$ | 0.2 |
| $N \rightarrow telescope$ | 0.7 |
| $N \rightarrow sandwich$ | 0.1 |
| $PN \rightarrow I$ | 1.0 |
| $V \rightarrow saw$ | 0.5 |
| $V \rightarrow ate$ | 0.5 |
| $P \rightarrow with$ | 0.6 |
| $P \rightarrow in$ | 0.4 |
| $D \rightarrow a$ | 0.3 |
| $D \rightarrow the$ | 0.7 |

# Probabilistic parsing

- We discussed the recognition problem:
  - check if a sentence is parsable with a CFG
- Now we consider parsing with PCFGs
  - Recognition with PCFGs: what is the probability of the most probable parse tree?
  - Parsing with PCFGs: What is the most probable parse tree?

# CFGs

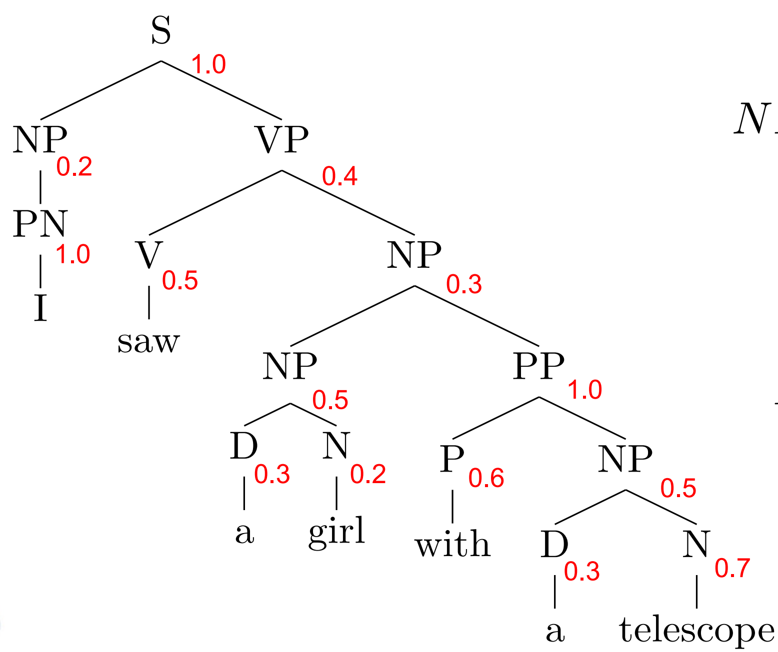
$S \rightarrow NP\ VP$ 1.0

$VP \rightarrow V$ 0.2
$VP \rightarrow V\ NP$ 0.4
$VP \rightarrow VP\ PP$ 0.4

$NP \rightarrow NP\ PP$ 0.3
$NP \rightarrow D\ N$ 0.5
$NP \rightarrow PN$ 0.2

$PP \rightarrow P\ NP$ 1.0

$N \rightarrow girl$ 0.2
$N \rightarrow telescope$ 0.7
$N \rightarrow sandwich$ 0.1
$PN \rightarrow I$ 1.0
$V \rightarrow saw$ 0.5
$V \rightarrow ate$ 0.5
$P \rightarrow with$ 0.6
$P \rightarrow in$ 0.4
$D \rightarrow a$ 0.3
$D \rightarrow the$ 0.7




$$p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times$$
$$0.5 \times 0.3 \times 0.2 \times 1.0 \times 0.6 \times 0.5 \times 0.3 \times 0.7$$
$$= 2.26 \times 10^{-5}$$

# Distribution over trees

- Let us denote by $G(x)$ the set of derivations for the sentence $x$
- The probability distribution defines the scoring $P(T)$ over the trees $T \in G(x)$
- Finding the best parse for the sentence according to PCFG:

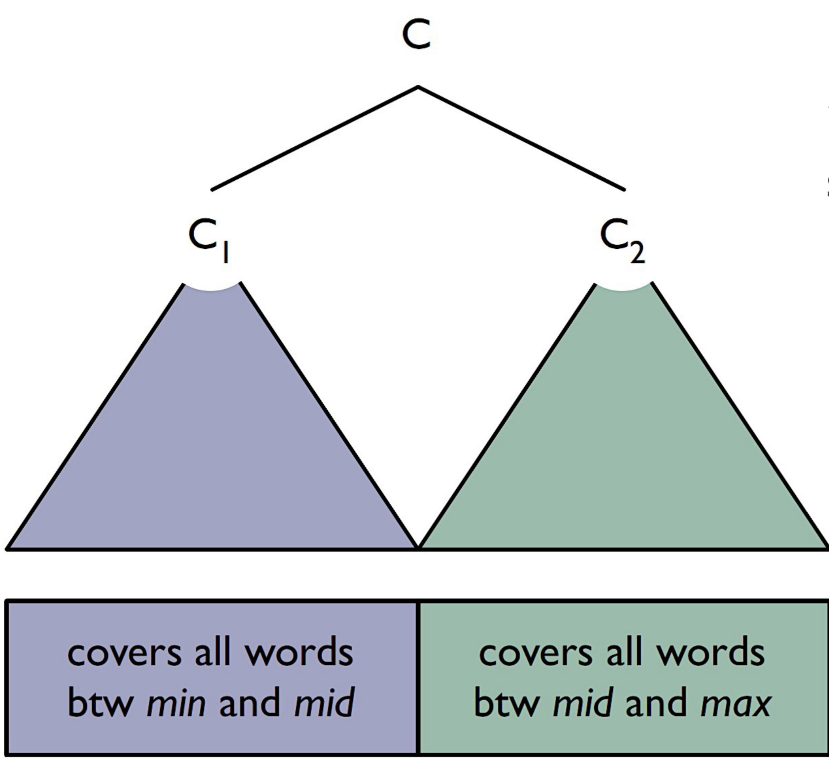
$$\arg\max_{T \in G(x)} P(T)$$

# CKY with PCFGs

- Chart is represented by a double array `chart[min][max][C]`
  - It stores probabilities for the most probable subtree with a given signature
- `chart[0][n][S]` will store the probability of the most probable full parse tree

# Intuition

$$C \to C_1 \;\; C_2$$




For every $C$ choose $C_1$, $C_2$ and mid such that

$$P(T_1) \times P(T_2) \times P(C \to C_1 C_2)$$

is maximal, where $T_1$ and $T_2$ are left and right subtrees.

# Implementation: preterminal rules

```
for each wi from left to right
  for each preterminal rule C -> wi
    chart[i - 1][i][C] = p(C -> wi)
```

# Implementation: binary rules

```
for each max from 2 to n
  for each min from max - 2 down to 0
    for each syntactic category C
      double best = undefined
      for each binary rule C -> C1 C2
        for each mid from min + 1 to max - 1
          double t1 = chart[min][mid][C1]
          double t2 = chart[mid][max][C2]
          double candidate = t1 * t2 * p(C -> C1 C2)
          if candidate > best then
            best = candidate
      chart[min][max][C] = best
```

# Unary rules

- Similarly to CFGs: after producing scores for signatures *(c, i, j)*, try applying unary rules (and rule chains)

# Unary (reflexive transitive) closure

$$\begin{array}{ll} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.2 \\ \ldots & \end{array} \quad \Rightarrow \quad \begin{array}{lr} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.2 \\ A \rightarrow C & 0.2 \times 0.1 \\ & \ldots \end{array} \quad \begin{array}{ll} A \rightarrow A & 1 \\ B \rightarrow B & 1 \\ C \rightarrow C & 1 \end{array}$$

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent

# Unary (reflexive transitive) clo

The fact that the rule is composite needs to be stored to recover the true tree

$$A \to B \quad 0.1$$
$$B \to C \quad 0.2$$
$$\dots$$

$$\Rightarrow$$

$$\begin{array}{lr} A \to B & 0.1 \\ B \to C & 0.2 \\ A \to C & 0.2 \times 0.1 \\ & \dots \end{array} \qquad \begin{array}{lr} A \to A & 1 \\ B \to B & 1 \\ C \to C & 1 \end{array}$$

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent

# Unary (reflexive transitive) clo

The fact that the rule is composite needs to be stored to recover the true tree

$$\begin{array}{lr} A \to B & 0.1 \\ B \to C & 0.2 \\ \dots & \end{array} \Rightarrow \begin{array}{lr} A \to B & 0.1 \\ B \to C & 0.2 \\ A \to C & 0.2 \times 0.1 \\ & \dots \end{array} \quad \begin{array}{lr} A \to A & 1 \\ B \to B & 1 \\ C \to C & 1 \end{array}$$

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent

$$\begin{array}{lr} A \to B & 0.1 \\ B \to C & 0.2 \\ A \to C & 1.e-5 \end{array} \Rightarrow \begin{array}{lr} A \to B & 0.1 \\ B \to C & 0.1 \\ A \to C & 0.02 \end{array} \quad \begin{array}{lr} A \to A & 1 \\ B \to B & 1 \\ C \to C & 1 \end{array}$$

# Unary (reflexive transitive) closure

The fact that the rule is composite needs to be stored to recover the true tree

$$\begin{array}{lll} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.2 \\ \ldots \end{array} \quad \Rightarrow \quad \begin{array}{lr} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.2 \\ A \rightarrow C & 0.2 \times 0.1 \\ & \ldots \end{array} \qquad \begin{array}{ll} A \rightarrow A & 1 \\ B \rightarrow B & 1 \\ C \rightarrow C & 1 \end{array}$$

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent

$$\begin{array}{lr} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.2 \\ A \rightarrow C & 1.e-5 \end{array} \quad \Rightarrow \quad \begin{array}{lr} A \rightarrow B & 0.1 \\ B \rightarrow C & 0.1 \\ A \rightarrow C & 0.02 \end{array} \qquad \begin{array}{ll} A \rightarrow A & 1 \\ B \rightarrow B & 1 \\ C \rightarrow C & 1 \end{array}$$

What about loops, like: $A \rightarrow B \rightarrow A \rightarrow C$ ?

# Recovery of the tree

- For each signature we store backpointers to the elements from which it was built (e.g., rule and, for binary rules, midpoint)
  - start recovering from *[0, n, S]*

- Be careful with unary rules
  - Basically you can assume that you always used an unary rule from the closure (but it could be the trivial one $C \rightarrow C$)

# Speeding up the algorithm (approximate search)

- Basic pruning (roughly):
  - For every span (i,j) store only labels which have the probability at most N times smaller than the probability of the most probable label for this span
  - Check not all rules but only rules yielding subtree labels having non-negligible probability
- Coarse-to-fine pruning
  - Parse with a smaller (simpler) grammar, and precompute (posterior) probabilities for each spans, and use only the ones with non-negligible probability from the previous grammar

# Parser evaluation

Though has many drawbacks it is easier and allows us to track state of the art across years

- Intrinsic evaluation:
  - Automatic: evaluate against annotation provided by human experts *(gold standard) according to some predefined measure*
  - Manual: … according to human judgment

- Extrinsic evaluation: score syntactic representation by comparing how well a system using this representation performs on some task
  - E.g., use syntactic representation as input for a semantic analyzer and compare results of the analyzer using syntax predicted by different parsers.

# Standard evaluation setting in parsing

- Automatic intrinsic evaluation is used: parsers are evaluated against gold standard by provided by linguists
- There is a standard split into the parts:
  - training set: used for estimation of model parameters
  - development set: used for tuning the model (initial experiments)
  - test set: final experiments to compare against previous work

# Automatic evaluation of constituent parsers

The most standard measure; we will focus on it

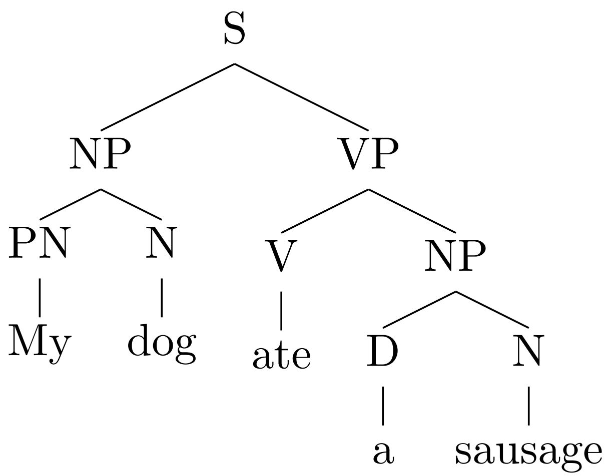
- Exact match: percentage of trees predicted correctly
- Bracket score: scores how well individual phrases (and their boundaries) are identified
- Crossing brackets: percentage of phrases boundaries crossing

# Brackets scores

= Subtree signatures for CKY

- The most standard score is bracket score
- It regards a tree as a collection of brackets: $[min, max, C]$
- The set of brackets predicted by a parser is compared against the set of brackets in the tree annotated by a linguist
- Precision, recall and F1 are used as scores

# Bracketing notation




- The same tree as a bracketed sequence

```
(S
   (NP  (PN My) (N Dog) )
   (VP  (V ate)
          (NP (D a )  (N sausage) )
   )
)
```

# Brackets scores

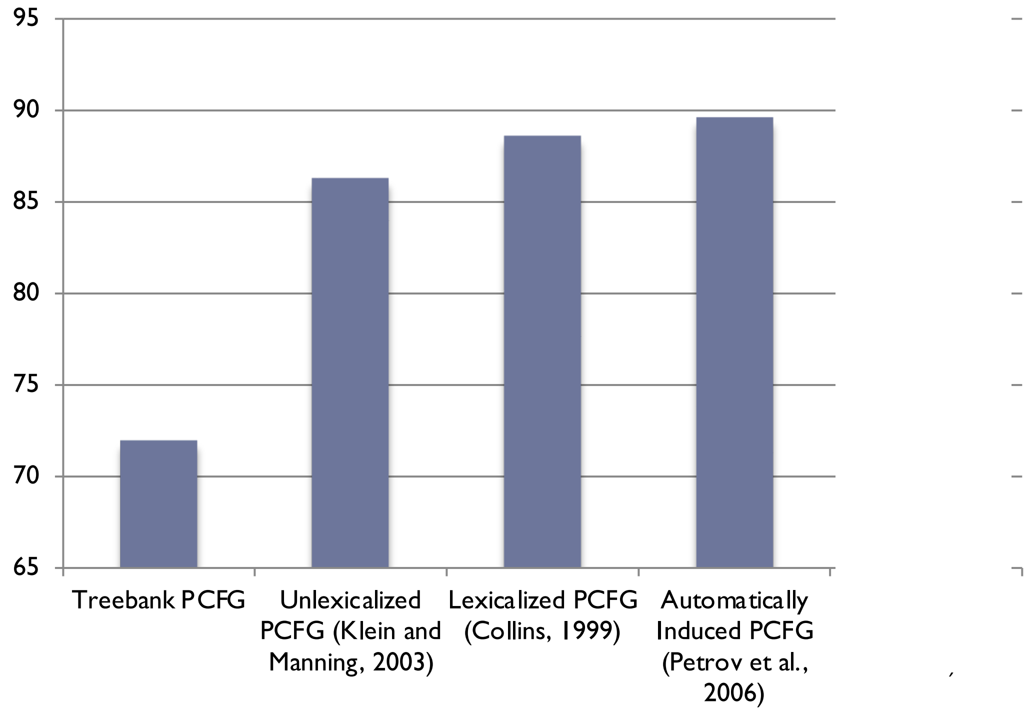
$$Pr = \frac{\text{number of brackets the parser and annotation agree on}}{\text{number of brackets predicted by the parser}}$$

$$Re = \frac{\text{number of brackets the parser and annotation agree on}}{\text{number of brackets in annotation}}$$

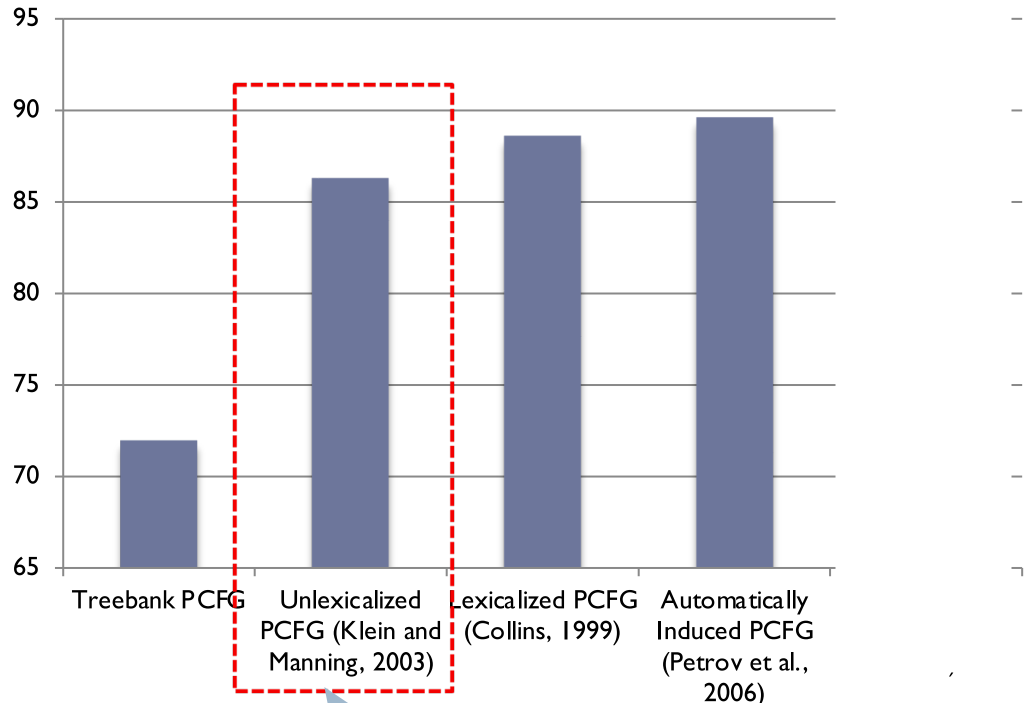
$$F1 = \frac{2 \times Pr \times Re}{Pr + Re}$$

Harmonic mean of precision and recall

# Preview: F1 bracket score



# Preview: F1 bracket score



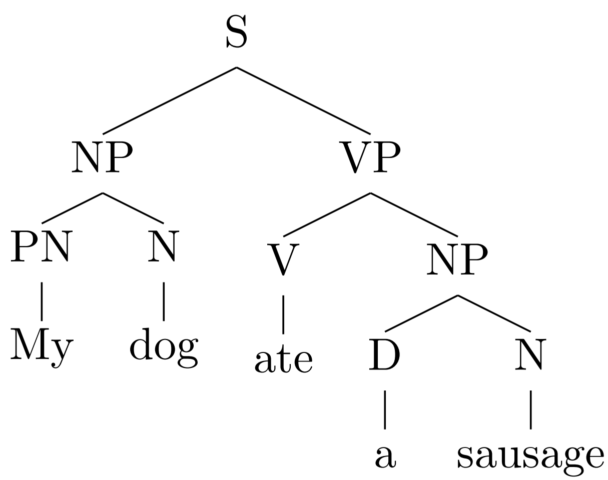

We will discuss how to achieve this

# Today

- Evaluation
- (Treebank) PCFG weaknesses
- PCFG extension: structural annotation

# Treebank PCFG

- Directly read-off rules from the treebank:




$$
\begin{array}{rl}
S \rightarrow NP\, VP & 1 \\
NP \rightarrow PN\ N & 1 \\
PN \rightarrow My & 1 \\
N \rightarrow Dog & 1 \\
VP \rightarrow V\ NP & 1 \\
NP \rightarrow D\ N & 1 \\
D \rightarrow a & 1 \\
N \rightarrow sausage & 1
\end{array}
$$

In practice, we binarized it (we discussed this last Friday)

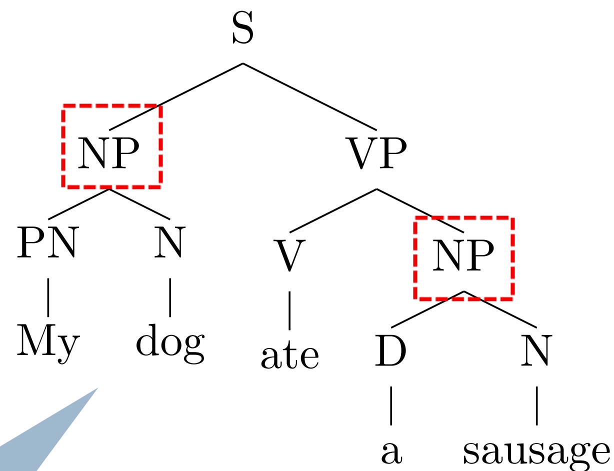
- The results are not great: around 72% F1

# Weaknesses of (treebank) PCFGs

- They do not encode lexical preferences
- They do not encode structural properties (beyond single rules)

# Context-free constraint

- Subject and object NPs are (statistically) very different
  - NPs under S vs. NPs under VP




Independence assumptions in PCFGs are too strong for this grammar

# Context-free constraint

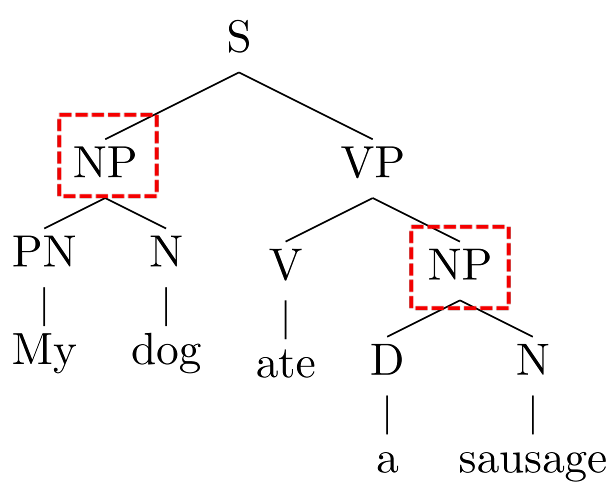
- Subject and object NPs are (statistically) very different
  - NPs under S vs. NPs under VP

| Types of NP | NP PP | D N | PN |
|---|---|---|---|
| All NPs | 11% | 9% | 6% |
| NPs under S (subjects) | 9% | 9% | 21% |
| NPs under VP (objects) | 23% | 7% | 4% |

Many more pronouns as subjects; prepositional phrases are much less frequent within subjects

# Context-free constraint

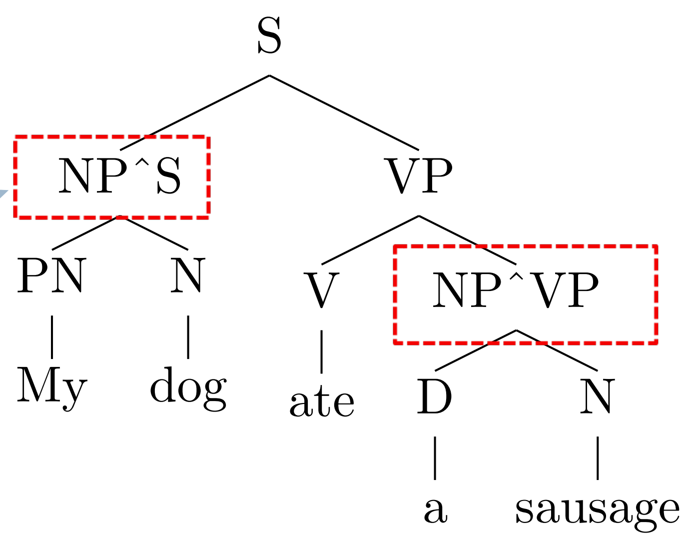
- Subject and object NPs are (statistically) very different
  - NPs under S vs. NPs under VP




How can we modify the grammar?

# Context-free constraint

- Subject and object NPs are (statistically) very different
  - NPs under S vs. NPs under VP

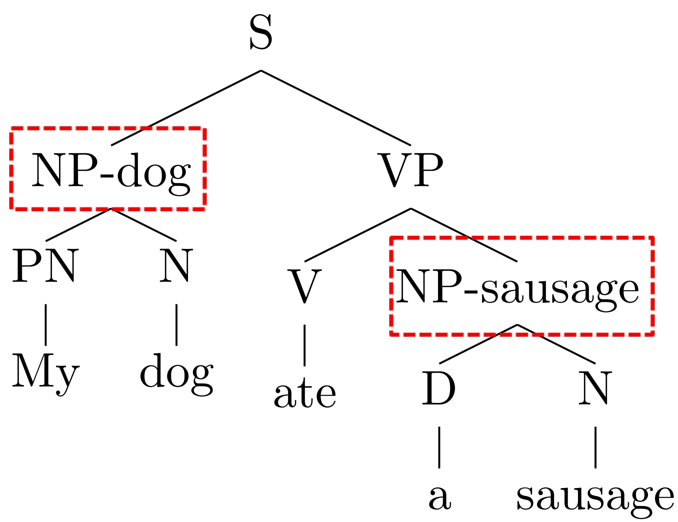



Structural annotation, specifically grandparent annotation *[Johnson 98]*

# Context-free constraint

- Subject and object NPs are (statistically) very different
  - NPs under S vs. NPs under VP

Lexicalization
*[Collins 99]*

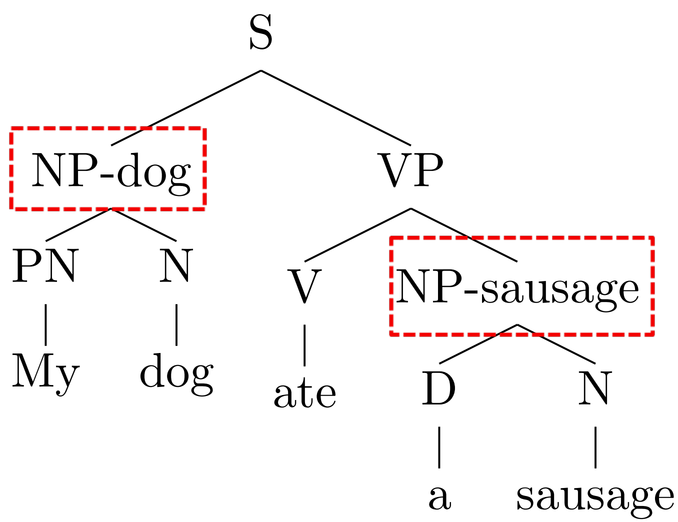


# Context-free constraint

We will get back to it tomorrow, as it is closely related to dep parsing

- Subject and object NPs are (statistically) very different
  - NPs under  S  vs.  NPs under VP

Lexicalization
*[Collins 99]*




Recall: instead of transforming the grammar we can see this in terms of transforming trees (on preprocessing) and then inducing a PCFG from the transformed treebank

# Today

- Evaluation
- (Treebank) PCFG weaknesses
- PCFG extension: structural annotation

# Approaches to enriching a grammar

- Structural annotation *[Johnson 98, Klein and Manning 03]*
- Lexicalization *[Collins 99, Charniak 00]*

Also known as grammar transforms

- There was a period in natural language processing when many researchers abandoned PCFGs and focused on richer modeling of context (history-based models) instead
- … but later research has showed that high accuracy can be achieved with PCFGs if an appropriate grammar is chosen

# Approaches to enriching a grammar

- Structural annotation *[Johnson 98, Klein and Manning 03]*
- Lexicalization *[Collins 99, Charniak 00]*

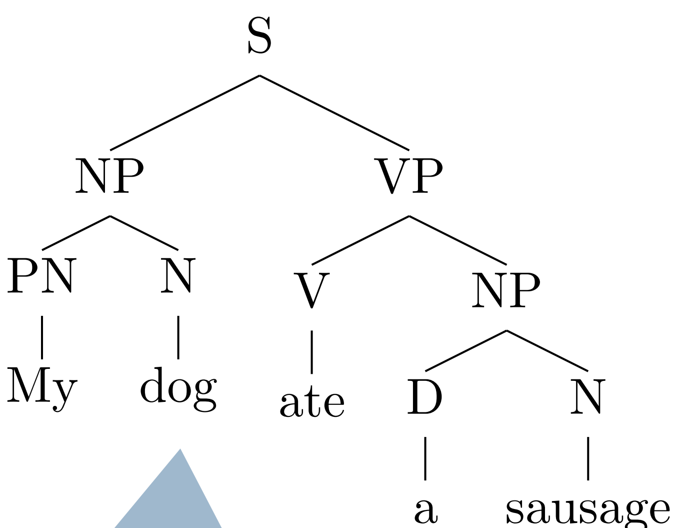
Also known as grammar transforms

- There was a period in natural language processing when many researchers abandoned PCFGs and focused on richer modeling of context (history-based models) instead
- … but later research has showed that high accuracy can be achieved with PCFGs if an appropriate grammar is chosen

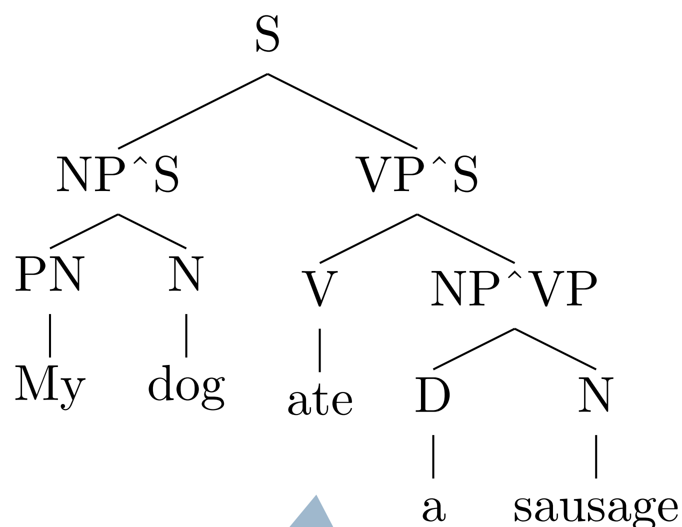
# Vertical Markovization

Recall, 1st and 2nd order HMMs, a similar idea

- Rule applications depend on past ancestors in the tree (not only parents) *[Johnson 98]*



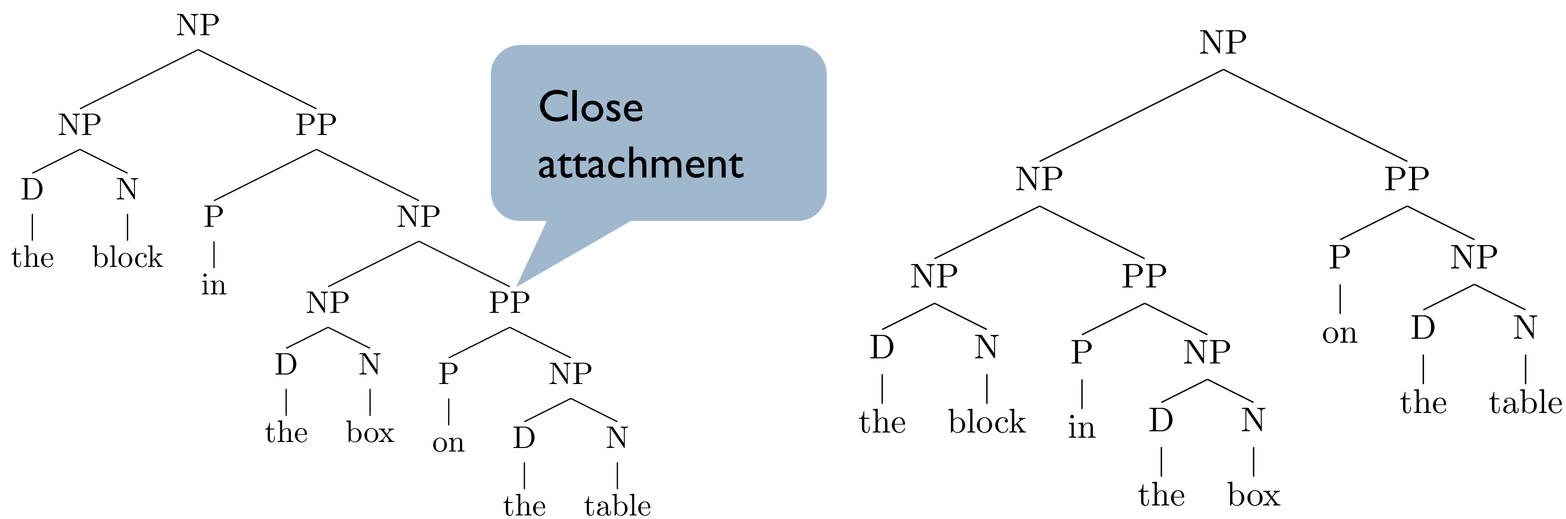

(Vertical) order 1




Vertical order 2

# PCFG weakness: Close Attachment

- Compare 2 configurations from a recent lecture:




- Close attachment is a-priori more likely (at least in Penn Treebank)
- Here they mean almost the same things (as the box in the box implies that box is also on the table)
  - … but consider:

San Jose cops kill man with knife

Text Paper Translate Listen Close

# San Jose cops kill man with knife

Ex-college football player, 23, shot 9 times allegedly charged police at fiancee's home

**By Hamed Aleaziz and Vivian Ho**

A man fatally shot by San Jose police officers while allegedly charging at them with a knife was a 23-year-old former football player at De Anza College in Cupertino who was distraught and depressed, his family said Thursday.

Police officials said two officers opened fire Wednesday afternoon on Phillip Watkins outside his fiancee's home because they feared for their lives. The officers had been drawn to the home, officials said, by a 911 call reporting an armed home invasion that, it turned out, had been made by Watkins himself.

But the mother of Watkins' fiancee, who also lives in the home on the 1300 block of Sherman Street, said she witnessed the shooting and described it as excessive. Faye Buchanan said the confrontation happened shortly after she called a suicide intervention hotline in hopes of getting Watkins medical help.

Watkins' 911 call came in at 5:01 p.m., said Sgt. Heather Randol, a San Jose police spokeswoman. "The caller stated there was a male breaking into his home armed with a knife," Randol said. "The caller also stated he was locked in an upstairs bedroom with his children and requested help from police."

She said Watkins was on the sidewalk in front of the home when two officers got there. He was holding a knife with a 4-inch blade and ran toward the officers in a threatening manner, Randol said.

"Both officers ordered the suspect to stop and drop the knife," Randol said. "The suspect continued to charge the officers with the knife in his hand. Both officers, fearing for their safety and defense of their life, fired at the suspect."

On the police radio, one officer said, "We have a male with a knife. He's walking toward us."

"Shots fired! Shots fired!" an officer said moments later.

A short time later, an officer reported, "Male is down. Knife's still in hand."

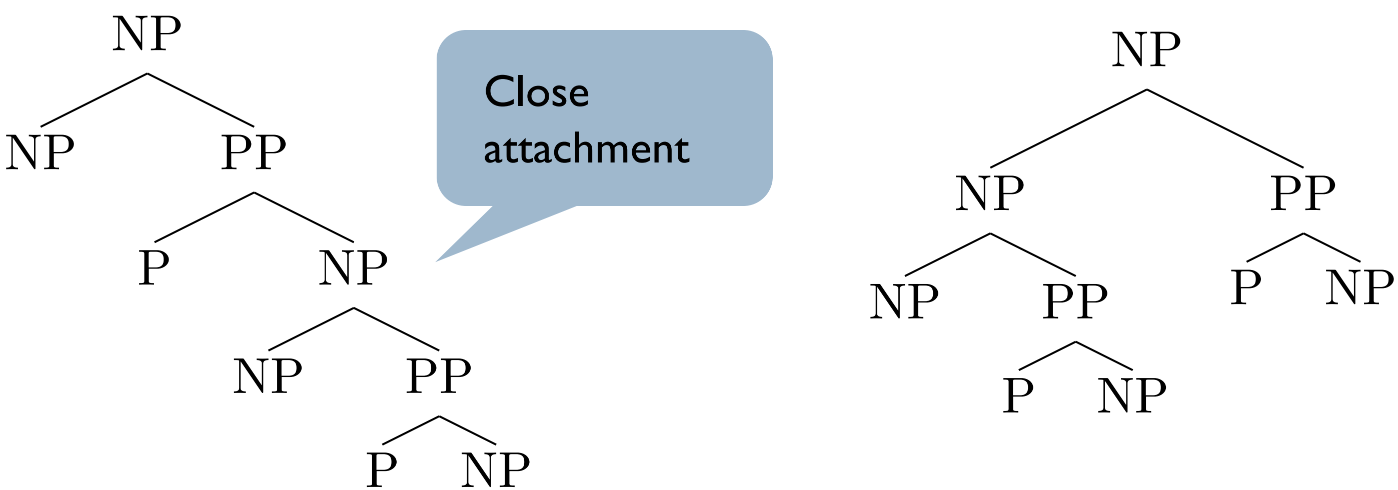
Buchanan said she had been prompted to call the

*Shoot continues on D8*

Back Continue

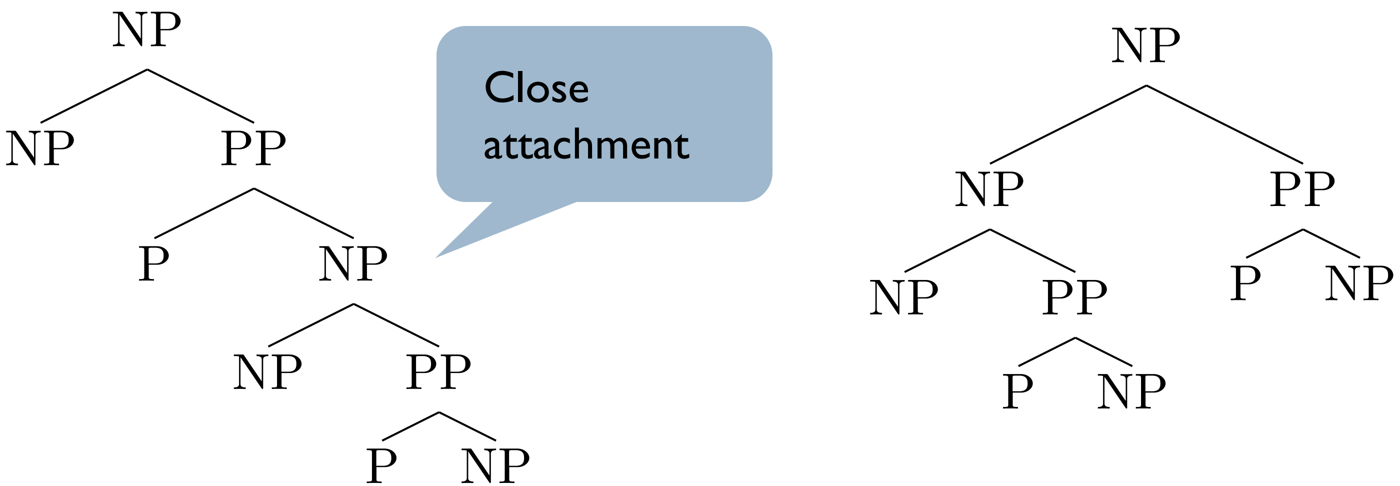
# PCFG weakness: Close Attachment

Can PCFG give a preference to one or another structure?



# PCFG weakness: Close Attachment

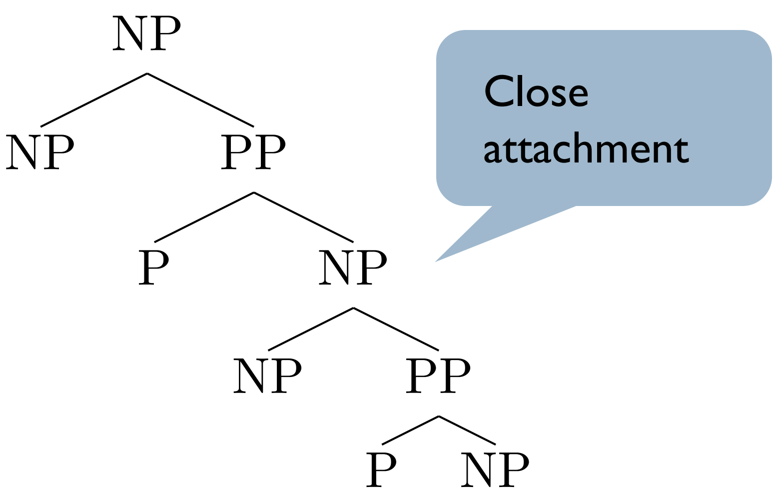
Can PCFG give a preference to one or another structure?

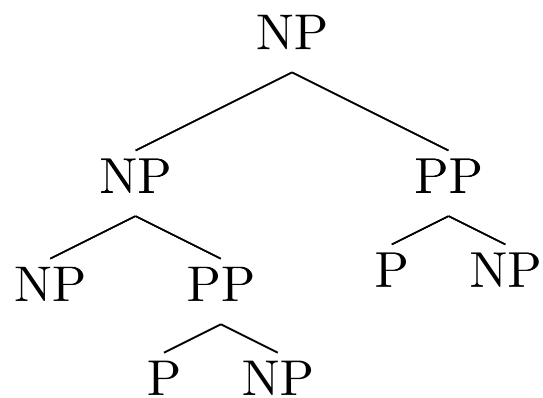



No, the same rules are used in both constructions, so a PCFG is guaranteed to return the same scores!

# PCFG weakness: Close Attachment

Can PCFG give a preference to one or another structure?

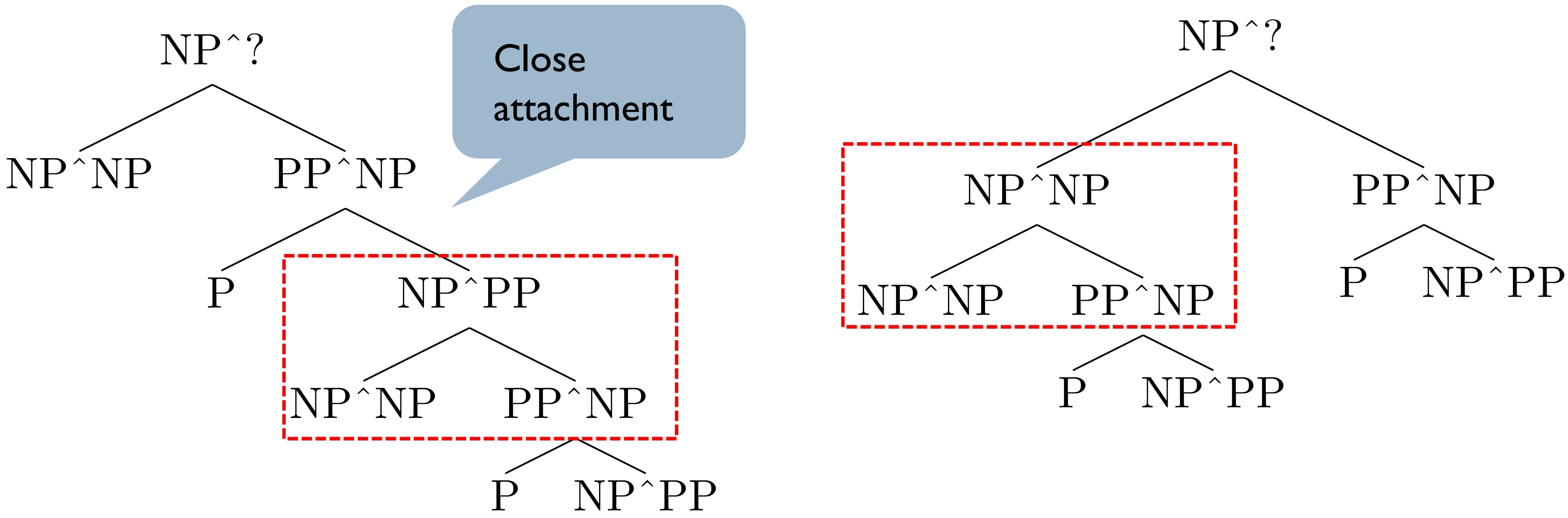



No, the same rules are used in both constructions, so a PCFG is guaranteed to return the same scores!

Would vertical Markovization help here (encode preference for close attachment)?

# PCFG weakness: Close Attachment




From the treebank, the enriched PCFG will assign higher probability to the rule

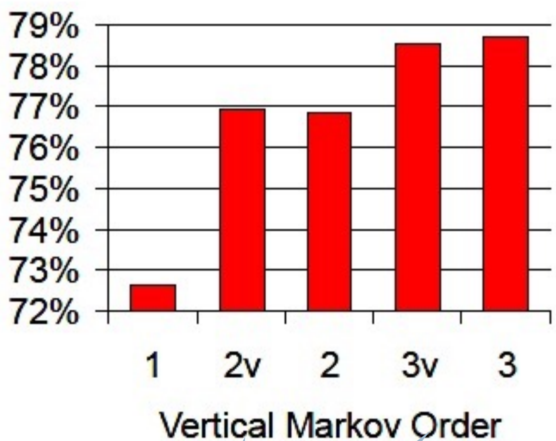
$$\text{NP\^{}}\textcolor{red}{\text{PP}} \rightarrow \text{NP\^{}NP} \quad \text{PP\^{}NP}$$

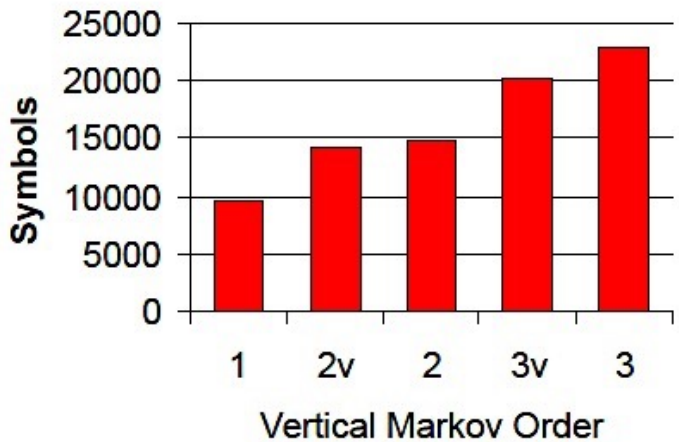
than to the rule

$$\text{NP\^{}}\textcolor{red}{\text{NP}} \rightarrow \text{NP\^{}NP} \quad \text{PP\^{}NP}$$

Consequently, higher accuracy (in average) is expected

# Vertical Markovization

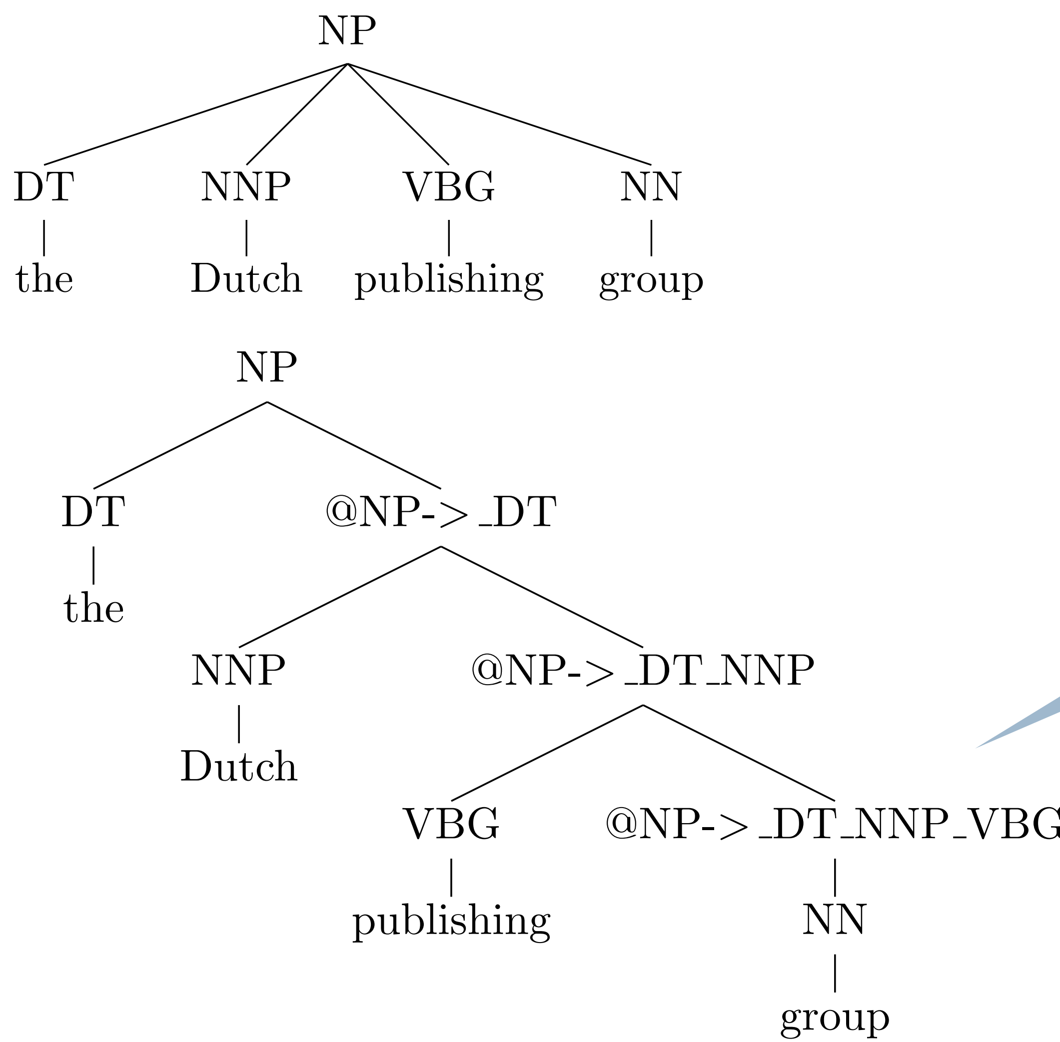



Selected histories

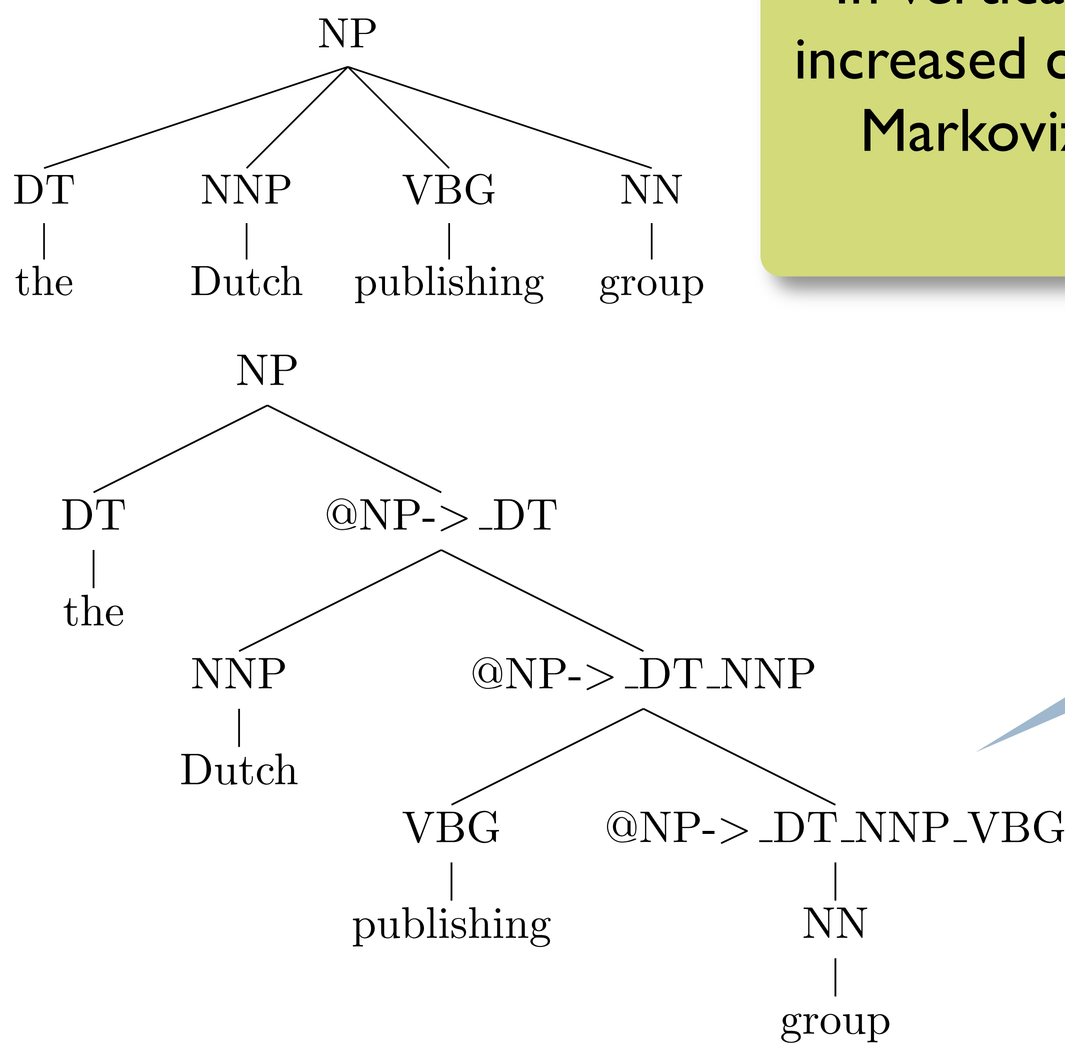
*[In this lecture some illustrations are adapted from Dan Klein]*

# Recall binarization (transformation to CNF form)




Can be regarded as horizontal history

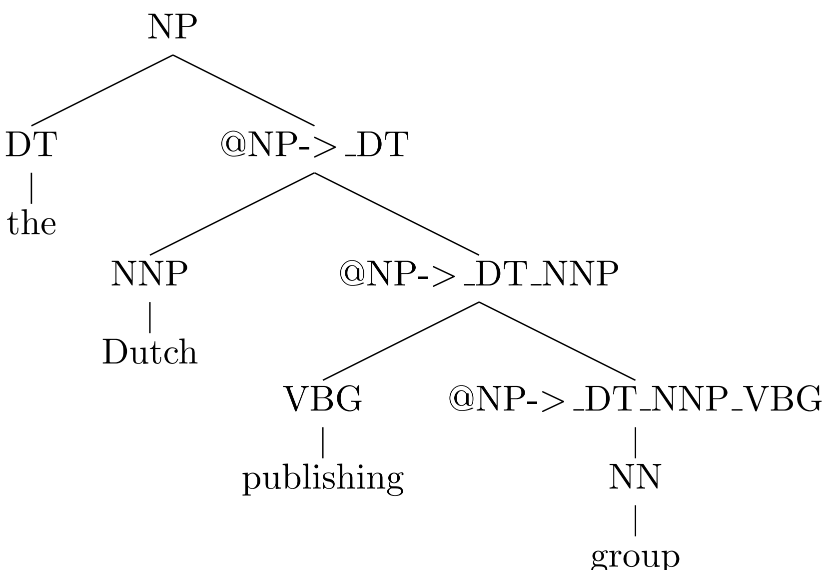
# Recall binarization (transformation to CNF form)



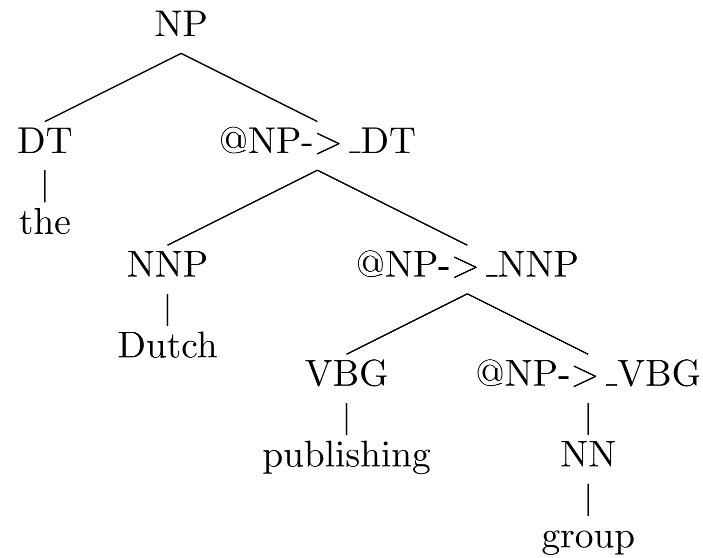

In vertical Markovization we increased context, in horizontal Markovization we want to reduce it

Can be regarded as horizontal history

# Recall binarization (transformation to CNF form)

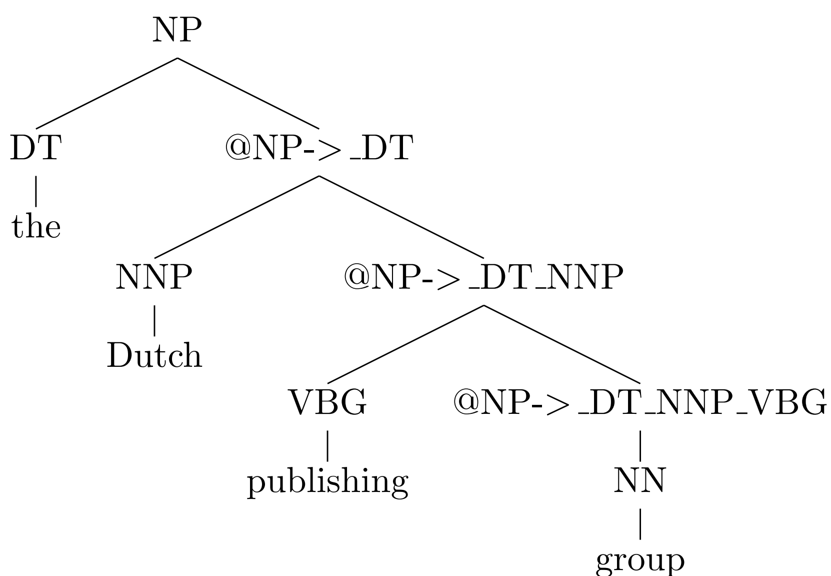


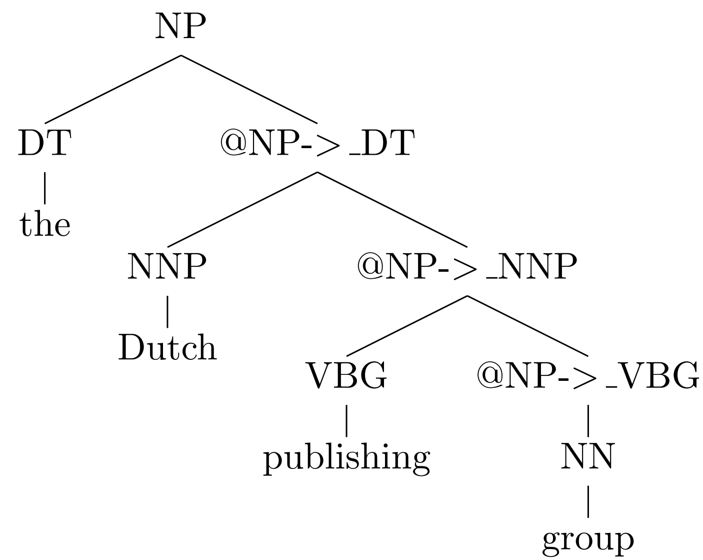

Horizontal order $h = \infty$




Horizontal order $h = 1$

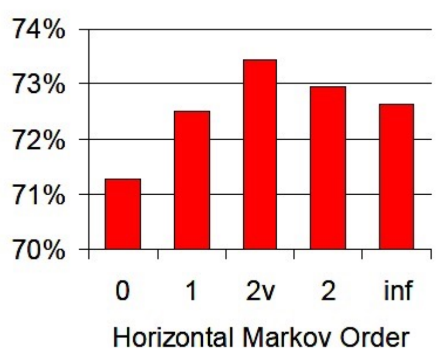
# Recall binarization (transformation to CNF form)

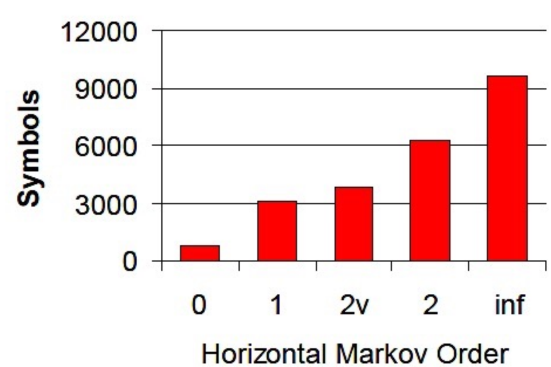


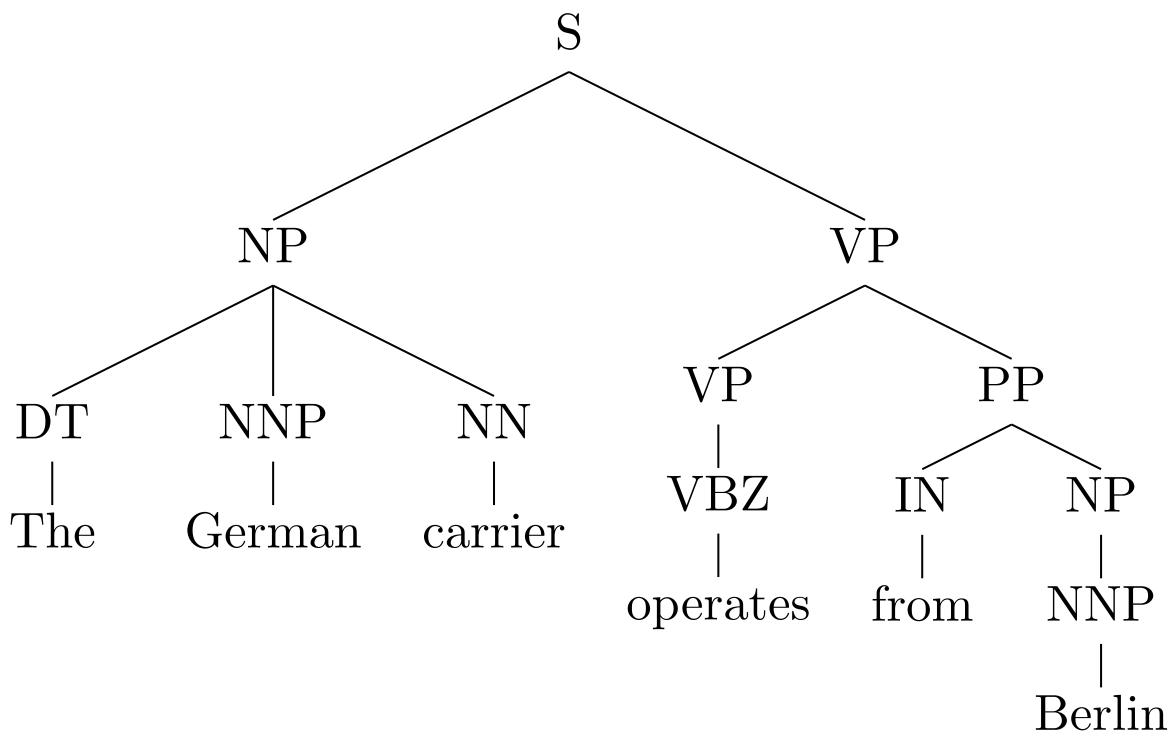

Horizontal order $h = \infty$

Horizontal order $h = 1$

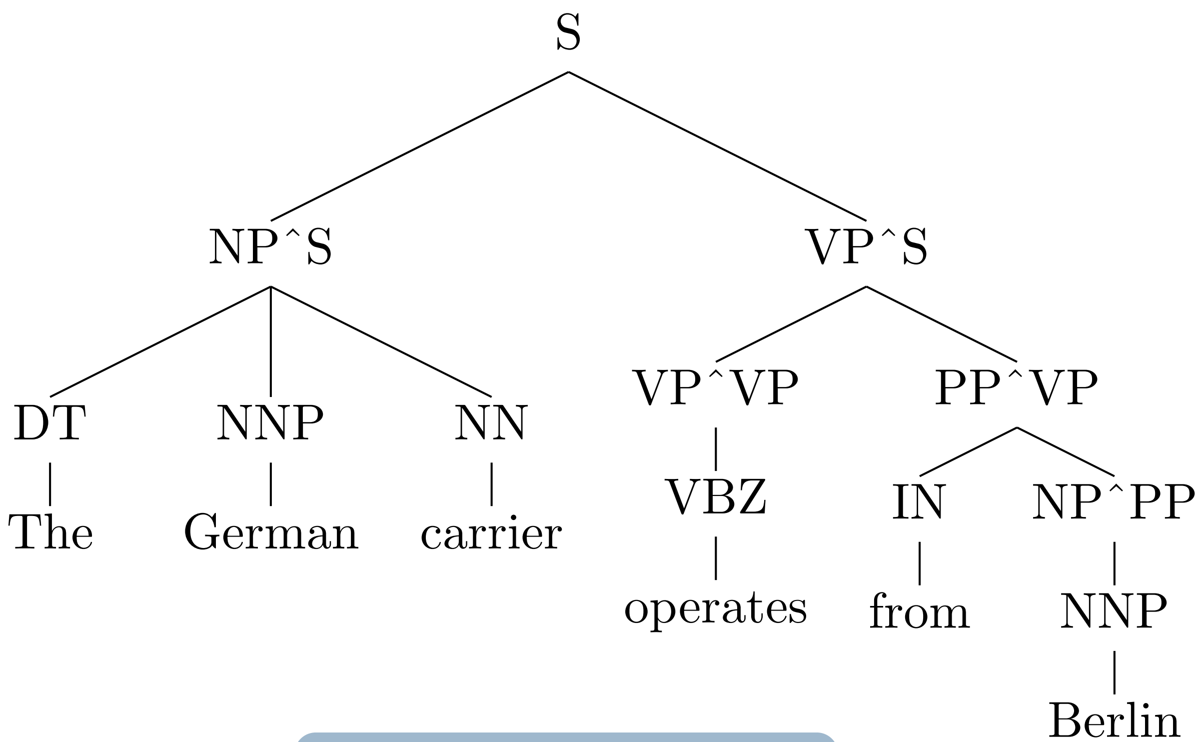


# Can we do both?

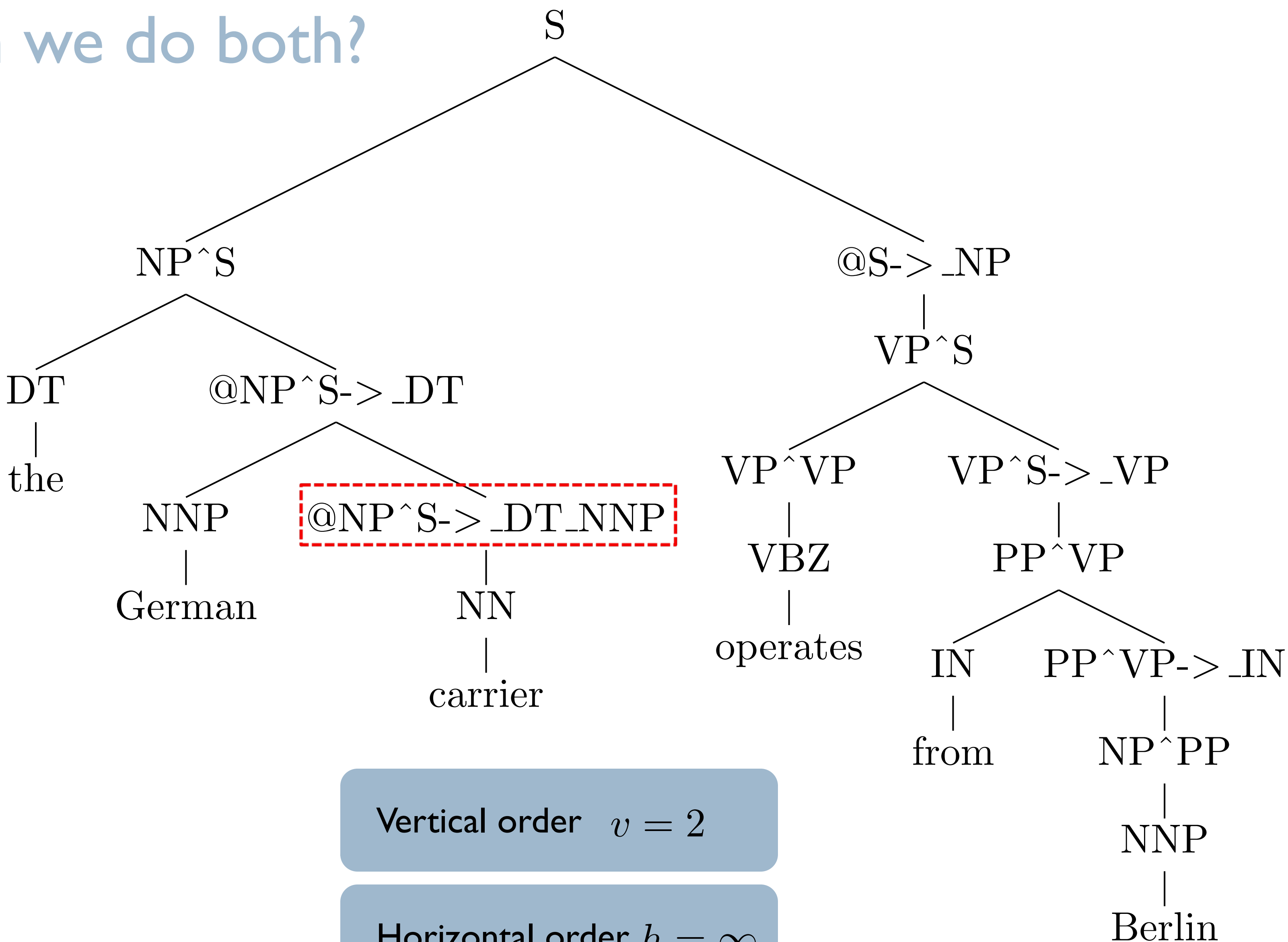


# Can we do both?




Vertical order $v = 2$

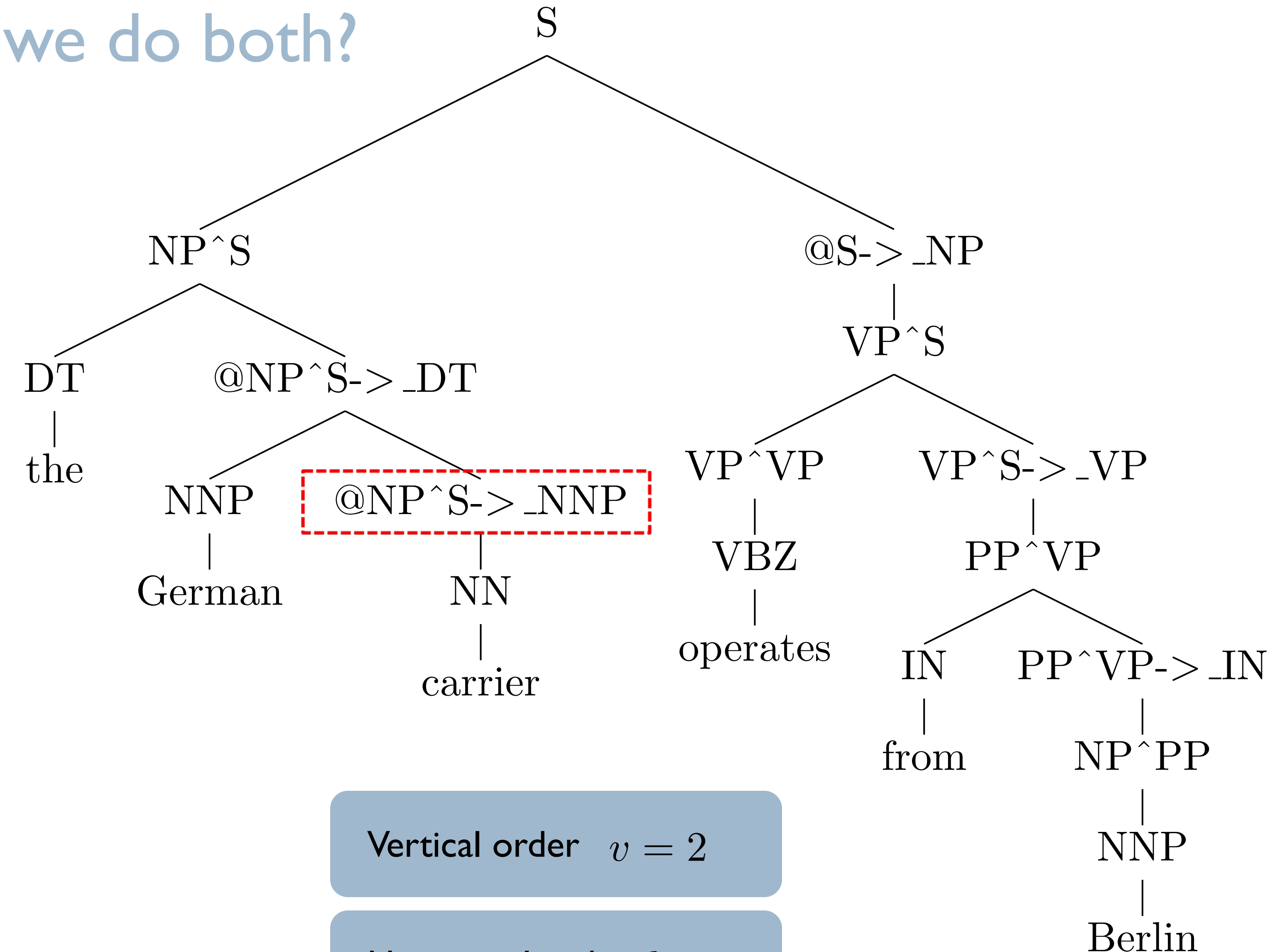
# Can we do both?




Vertical order $v = 2$

Horizontal order $h = \infty$

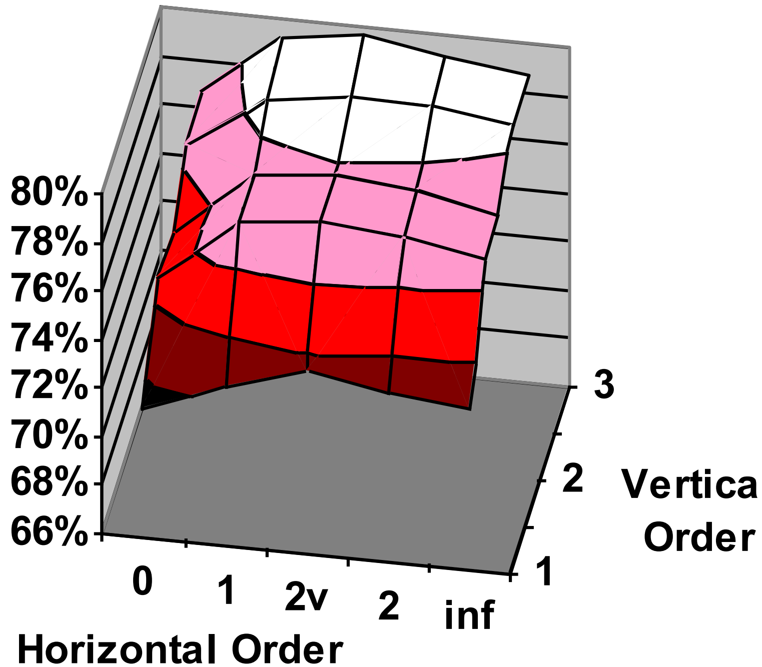
# Can we do both?




Vertical order $v = 2$

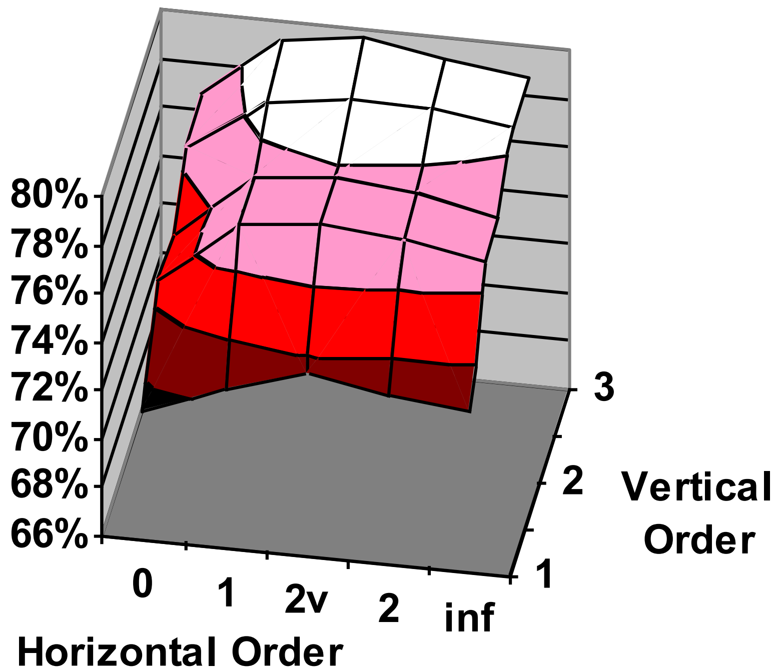
Horizontal order $h = 1$

# Vertical and Horizontal Markovization




Around 78%, compare with 72% for the original treebank PCFG

# Vertical and Horizontal Markovization




Around 78%, compare with 72% for the original treebank PCFG

Any idea how we can improve this using techniques we discussed?

# Splitting: PoS tags

- PoS tags in Penn Treebank are too coarse
- Very obvious for IN tag:
  - Assigned both to 'normal' prepositions (to form a prepositional phrase) – *in, on, at*, … –
  - and to subordinating conjunctions (e.g., *if*)
    - *E.g., check if advertising works*
- This change alone leads to a 2% boost in performance:
  - from 78.2 to 80.3

*[Klein and Manning 2003]*

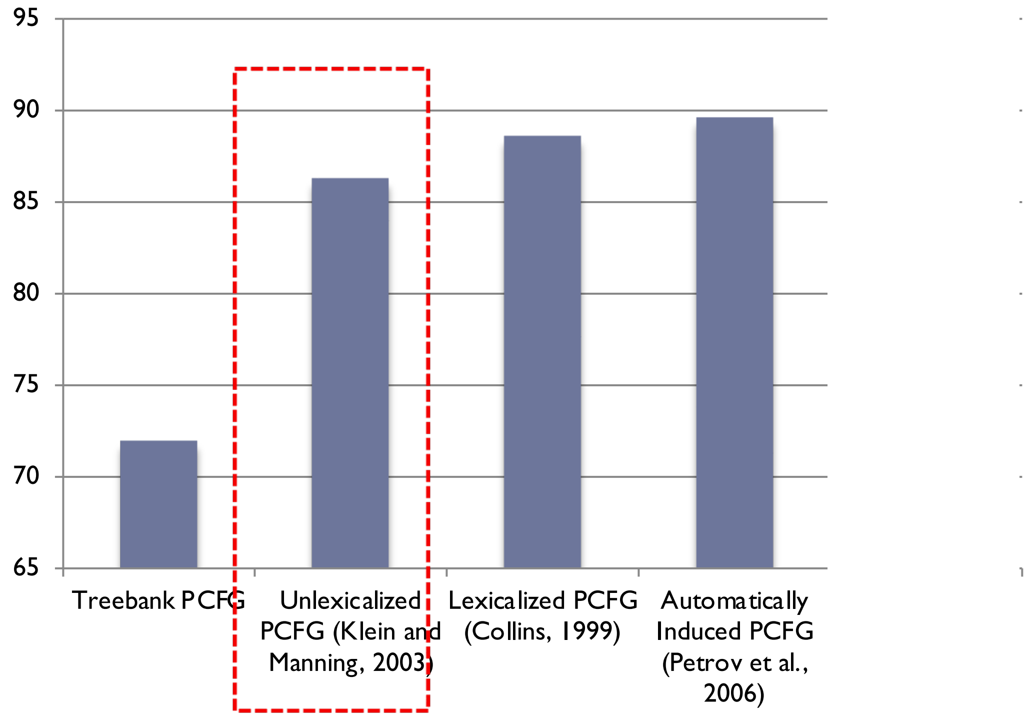
# Splitting: other symbols

- Split determiners: on demonstrative ("those") and others (e.g., "the", "a")
- Split adverbials: on phrasal and not ("quickly" vs. "very")
- …

All these changes (and a couple of other ones) lead to 86.3 % F1, a very respectable (and maybe even surprising) performance for an unlexicalized PCFG model

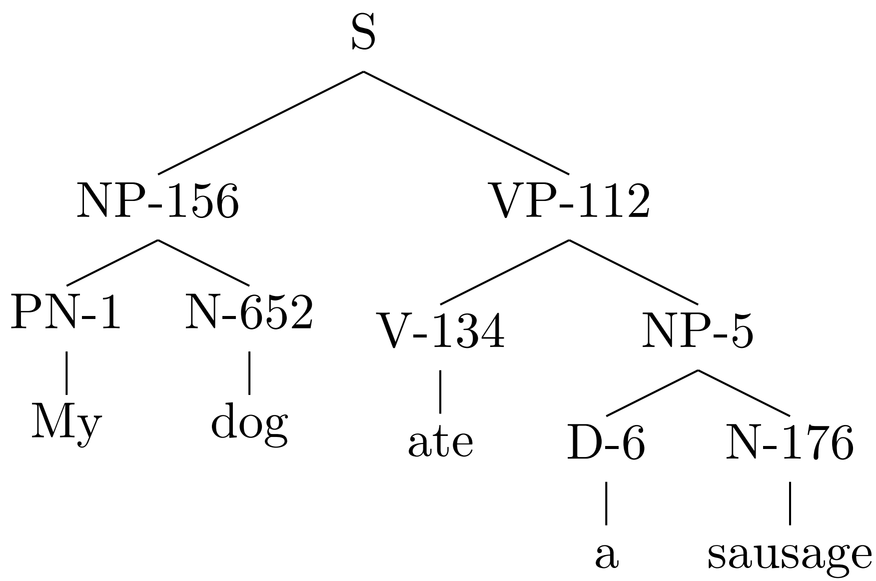
*[Klein and Manning 2003]*

# Preview: F1 bracket score



# Alternative ideas: inducing splits (through EM)

- Learning types of nonterminals from data, i.e. automatically enriching the grammar (Latent-annotated PCFGs, LA-PCFG)
  - One can think of this as a type of clustering of tree contexts of non-terminal symbols




**Around 90% F1**

*[Matsuzaki et al., 2005, Petrov et al., 2006]*

# Alternative ideas: anchored rules

- A rule probability is not constant but predicting for a given span in the chart
  - E.g., a neural network predicts the probability of a rule for a specific operation of the chart

```
double t1 = chart[min][mid][C1]

double t2 = chart[mid][max][C2]

double candidate = t1 * t2 * p(C -> C1 C2)
```

Instead use:

$$t_1 * t_2 * NN_\theta(C_1, C_2, C_3, min, max, mid, \mathbf{x})$$

Up to 97% F1

*First in Cross and Huang (2016)*

# Summary

- PCFGs for statistical parsing
- Dynamic programming algorithm for parsing with PCFGs
- Vanilla treebank PCFGs parser is (very) weak
- … but can be improved to produce a very strong system
- CKY is an important tool, used in many applications