Maschinelle Sprachverarbeitung

Parsing with Probabilistic Context-Free Grammar

Ulf Leser

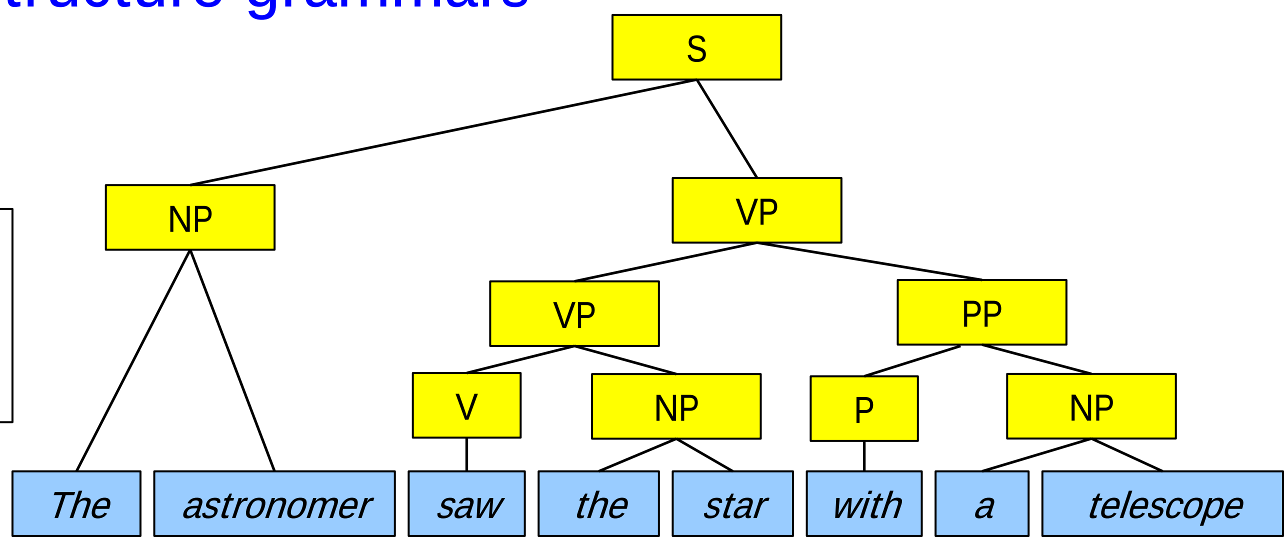
Content of this Lecture

- Phrase-Structure Parse Trees
- Probabilistic Context-Free Grammars
- Parsing with PCFG
- Other Issues in Parsing

Parsing Sentences

- POS tagging studies the plain sequence of words in a sentence
- But sentences have more and non-consecutive structures
- Plenty of linguistic theories exist about the nature and representation of these structures / units / phrases / ...
- Here: Phrase structure grammars

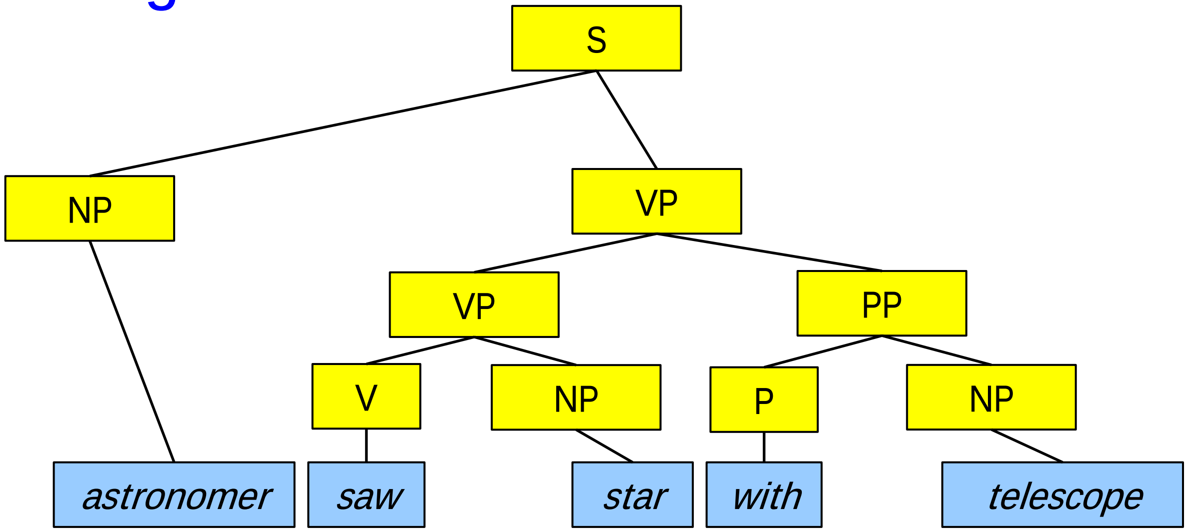
The astronomer saw the star with a telescope



Parsing Sentences

- POS tagging studies the plain sequence of words in a sentence
- But sentences have more and non-consecutive structures
- Plenty of linguistic theories exist about the nature and representation of these structures / units / phrases / ...
- Here: Phrase structure grammars

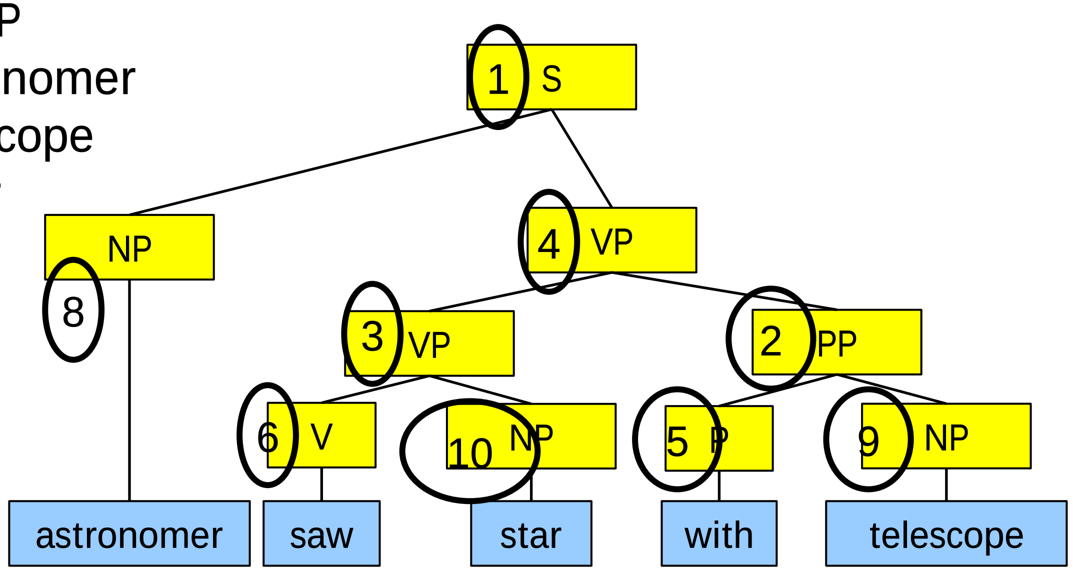
The astronomer saw the star with a telescope



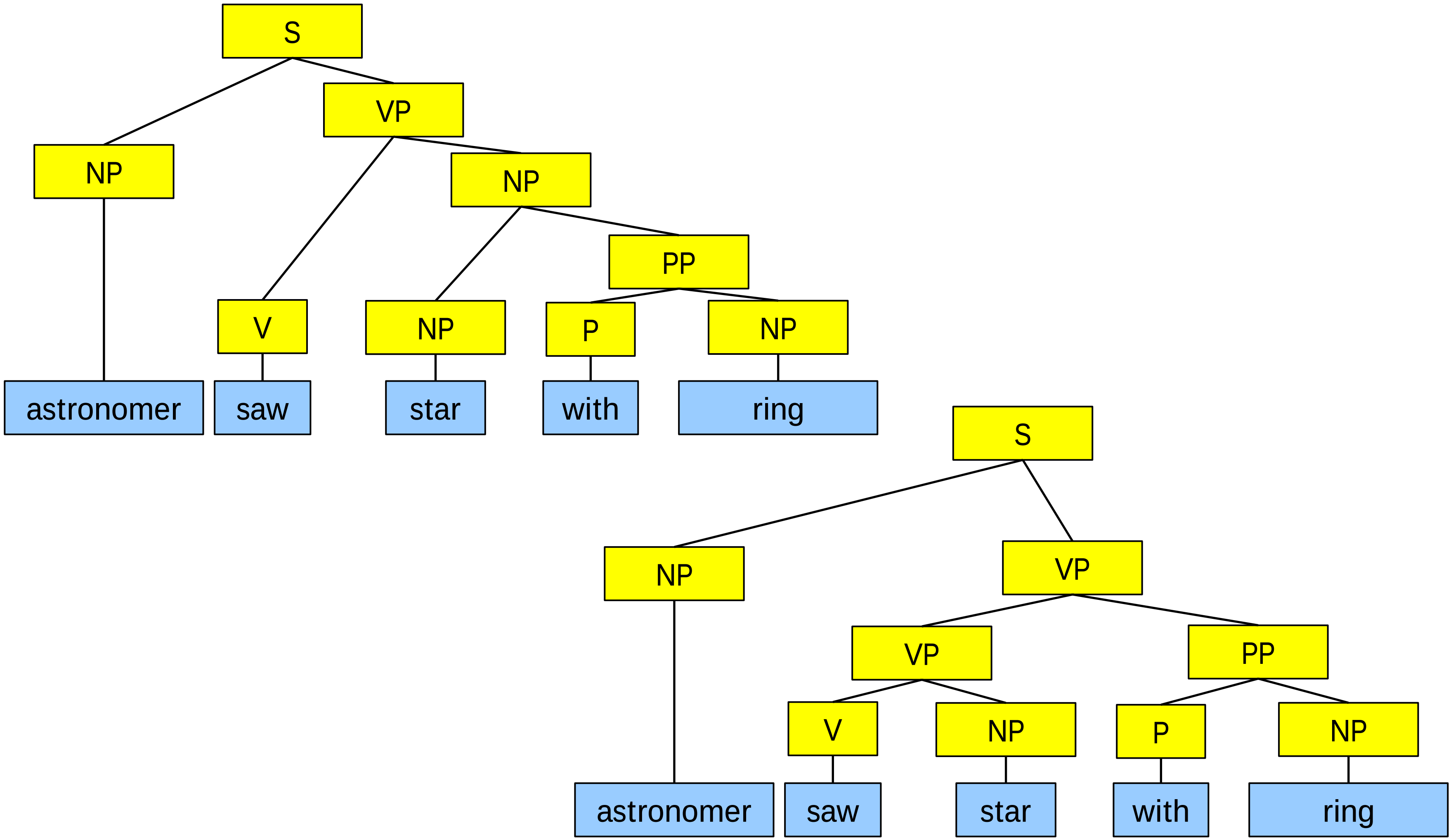
Phrase Structure Grammar

- Builds on assumptions
  - Sentences consist of nested structures
  - There is a fixed set of different structures (phrase types)
  - Nesting can be described by a context-free grammar

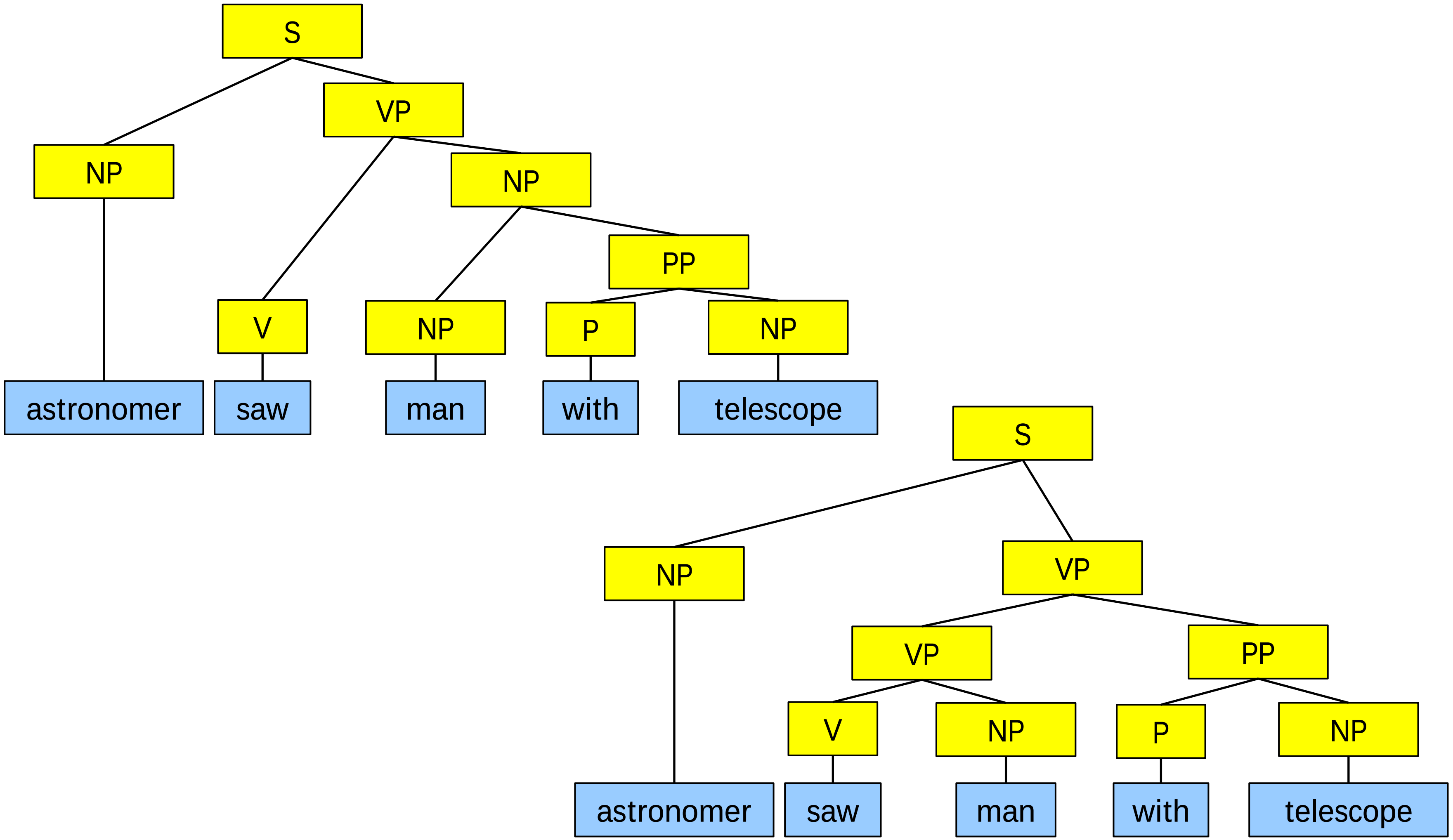
1: S → NP VP
2: PP → P NP
3: VP → V NP
4: VP → VP PP
5: P → with
6: V → saw
7: NP → NP PP
8: NP → astronomer
9: NP → telescope
10: NP → star



Ambiguity?


S
VP
NP
NP
PP
V
NP
P
NP
astronomer
saw
star
with
ring
S
NP
VP
VP
PP
V
NP
P
NP
astronomer
saw
star
with
ring

Problem 1: Ambiguity!

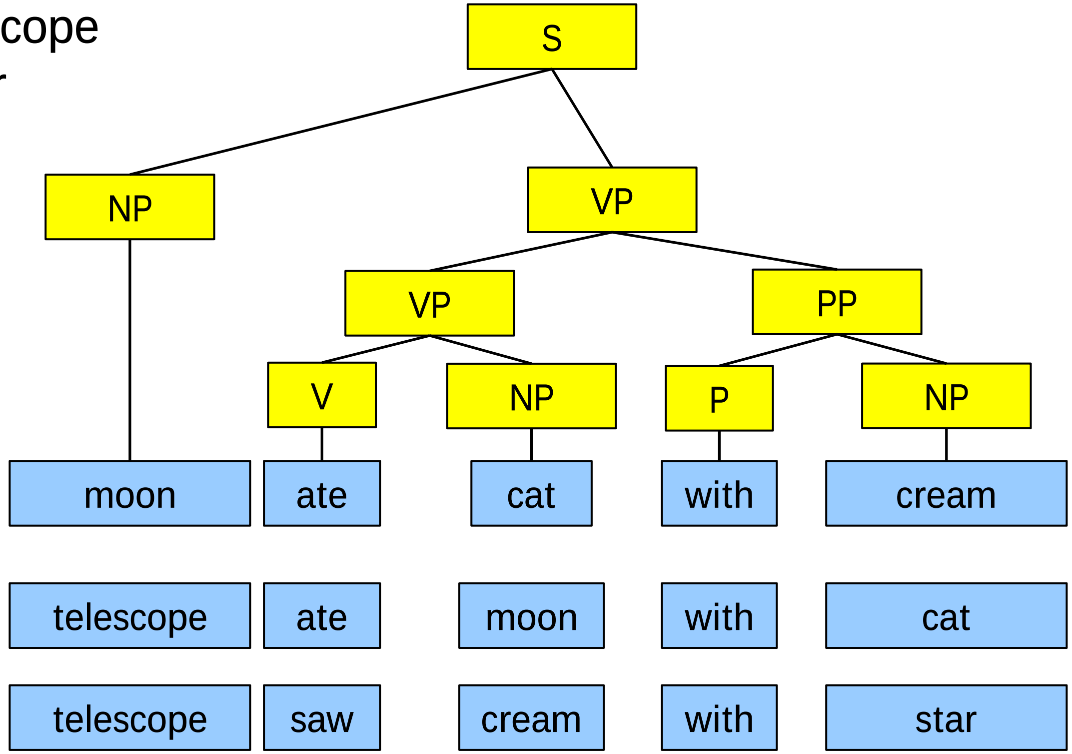


Problem 2: Syntax versus Semantics

- Phrase structure grammars only capture syntax

1: S → NP VP
2: PP → P NP
3: VP → V NP
4: VP → VP PP
5: P → with
6: V → saw
7: NP → NP PP
8: NP → astronomer
9: NP → telescope
10: NP → star

V → ate
NP → moon
NP → cat
NP → cream



Content of this Lecture

- Phrase-Structure Parse Trees
- Probabilistic Context-Free Grammars
- Parsing with PCFG
- Other Issues in Parsing

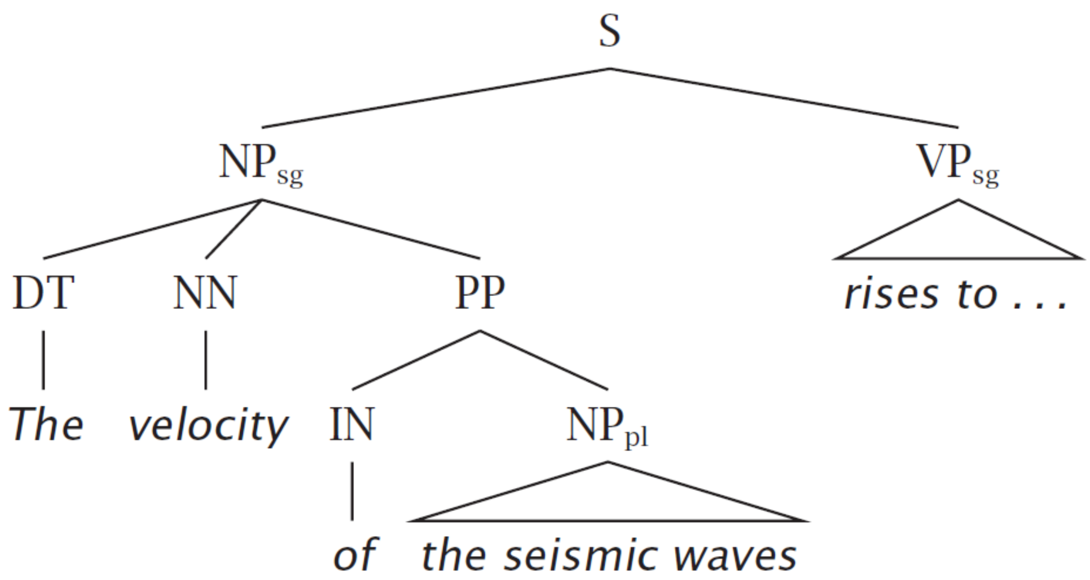
Probabilistic Context-Free Grammars (PCFG)

- Also called Stochastic Context Free Grammars
- Idea: Context free grammars by transition probabilities
  - Every rule gets a non-zero probability of firing
  - Grammar still recognizes the same language
  - But every parse can be assigned a probability
- Usages
  - Find parse with highest probability (“true” meaning)
  - Detect ambiguous sentences (>1 parses with similar probability)
  - What is the overall probability of a sentence given a grammar
    - Sum of the probabilities of all derivations producing the sentence
  - Language models: Predict most probable next token in an incomplete sentence which is allowed by the grammar rules

POS Tagging versus Parsing

- The velocity of the seismic waves rises to ...

- Difficult for a POS tagger: waves/Plural rises/Singular
- Simple for a PCFG



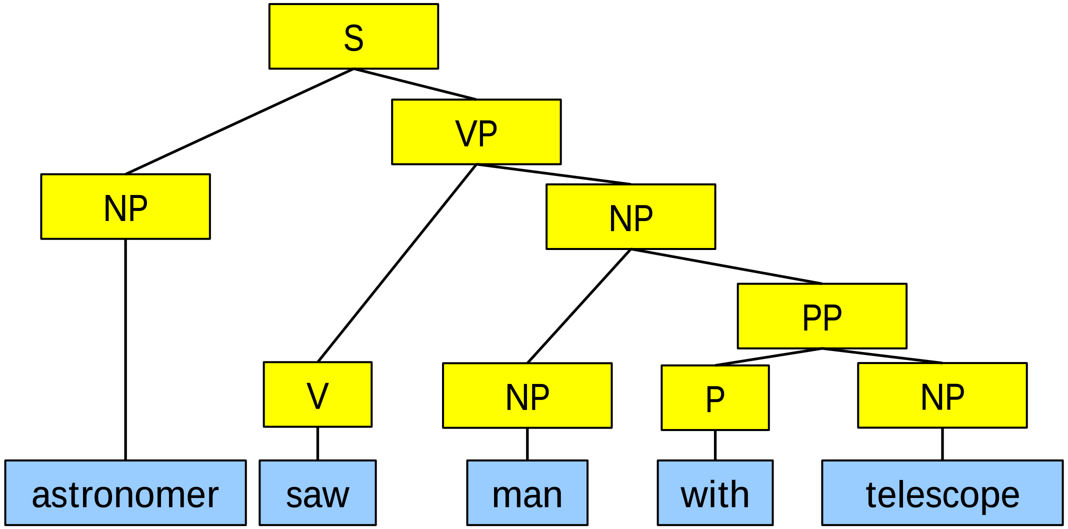
More Formal

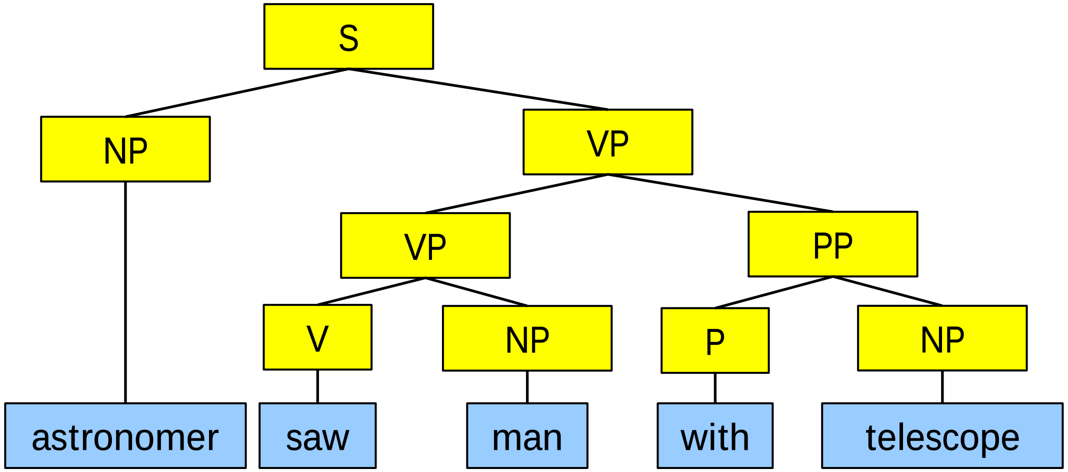
- Definition
  A PCFG if a 5-tuple (W, N, S, R, p) with
  - *W is a set of terminals (words)* $w_1, w_2, \ldots$
  - *N is a set of non-terminals (phrase types)* $N_1, N_2, \ldots$
  - *S is a designated start symbol*
  - *R is a set of rules* $\langle N_i \rightarrow \varphi \rangle$
    - *where* φ *is a sequence of terminals and or non terminals*
  - *p is a function assigning a non-zero probability to every rule such that*

$$\forall i: \sum_j p\left(N_i \rightarrow \varphi_j\right) = 1$$

Example

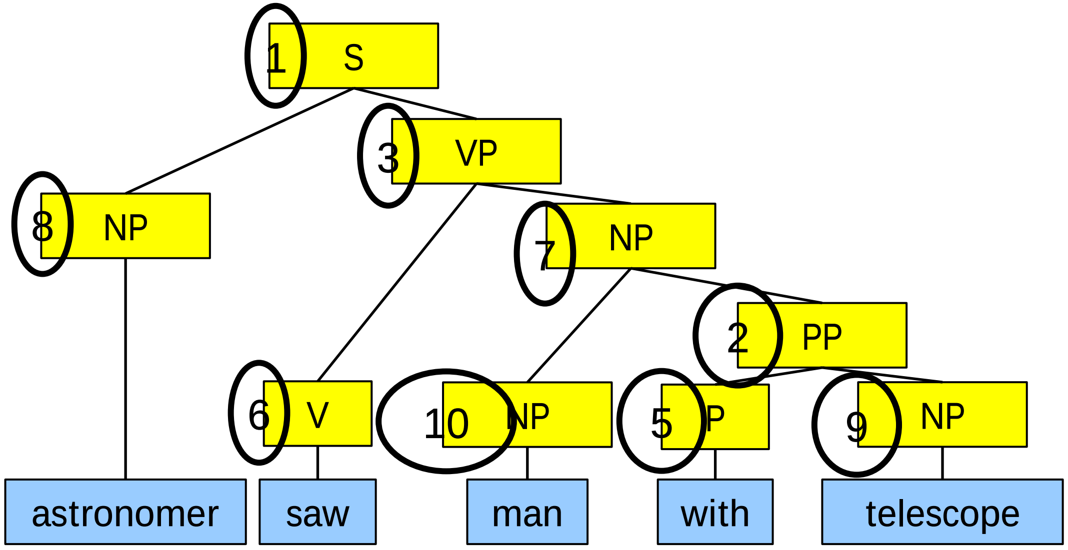
Rules	p
1: S → NP VP	1,00
2: PP → P NP	1,00
3: VP → V NP	0,30
4: VP → VP PP	0,70
5: P → with	1,00
6: V → saw	1,00
7: NP → NP PP	0,80
8: NP → astronomer	0,10
9: NP → telescope	0,05
10: NP → man	0,05





Example

1: S → NP VP 1,00
2: PP → P NP 1,00
3: VP → V NP 0,30
4: VP → VP PP 0,70
5: P → with 1,00
6: V → saw 1,00
7: NP → NP PP 0,80
8: NP → astronomer 0,10
9: NP → telescope 0,05
10: NP → man 0,05

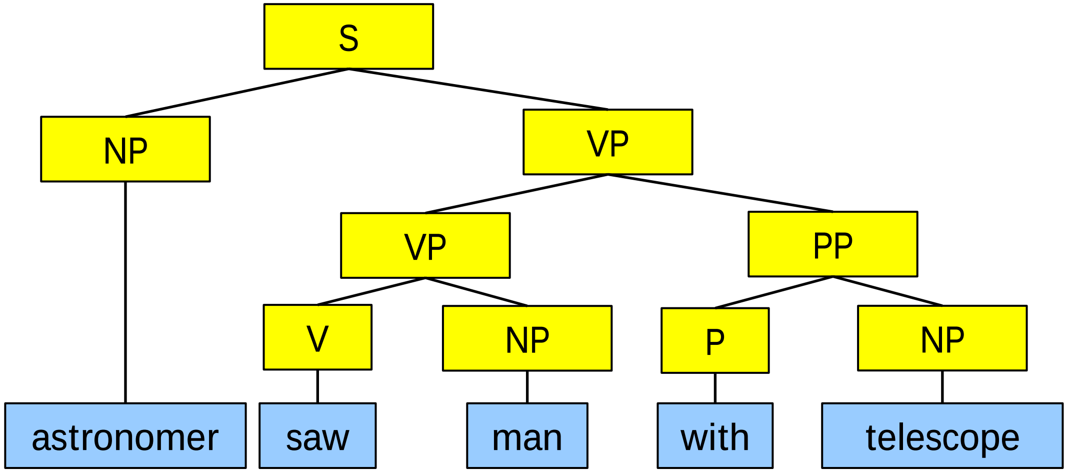


$p(t_1) = 1 * 0{,}1 * 0{,}3 * 1 * 0{,}8 * 0{,}05 * 1 * 1 * 0{,}05 = 0{,}0006$

Example

1: S → NP VP	1,00
2: PP → P NP	1,00
3: VP → V NP	0,30
4: VP → VP PP	0,70
5: P → with	1,00
6: V → saw	1,00
7: NP → NP PP	0,80
8: NP → astronomer	0,10
9: NP → telescope	0,05
10: NP → man	0,05

$p(t_2) = 1*0,1*0,7*0,3*1*0,05*1*1*0,05 = 0,000525$



Implicit Assumptions

- Context-free: Probability of a derivation of a subtree under non-terminal N is independent of anything else in the tree
  - Above N, left of N, right of N
- Place-invariant: Probability of a given rule r is the same anywhere in the tree
  - Probability of a subtree is independent of its position in the sentence
- Semantic-unaware: Probability of terminals do not depend on the co-occurring terminals in the sentence
  - Semantic validity is not considered

Usefulness (of a good PCFG)

- Tri-gram models are the better language models
  - Work at word level – conditional probabilities of word sequences
- PCFG are a step towards resolving ambiguity, but not a solution due to lack of semantics
- PCFG can produce robust parsers
  - When learned on a corpus with a few, rare errors, these are cast into rules with low probability
- Have some implicit bias (work-arounds known)
  - E.g. small trees get higher probabilities
- State-of-the-art parser combine PCFG with additional formalized knowledge

Three Issues

- Given a PCFG G and a sentence $s \in L(G)$
  - Issue 1: Decoding (or parsing): Which chain of rules (derivation) from G produced s with the highest probability?
  - Issue 2: Evaluation: What is the overall probability of s given G?
- Given a context free grammar G' and a set of sentences S with their derivation in G'
  - Issue 3: Learning: Which PCFG G with the same rule set as G' produces S with the highest probability?
  - We make our life simple: (1) G' is given, (2) sentences are parsed
  - Removing assumption (2) leads to an EM algorithm, removing (1) is hard (structure learning)
- Very close relationship to the same problems in HMMs

Chomsky Normal Form

- We only consider PCFG with rules of the following form (Chomsky Normal Form, CNF)
  - N $\rightarrow$ w   Non-terminal to terminal
  - N $\rightarrow$ N' N''   Non-terminal to two non terminals
  - Note: For any CFG G, there exists a CFG G' in Chomsky Normal Form such that G and G' are weakly equivalent, i.e., accept the same language (but with different derivations)

- Accordingly, a PCFG in CNF has $|N|^3+|N|*|W|$ parameter

Issue 3: Learning

- Given a context free grammar G' and a set of sentences S with their derivations in G': Which PCFG G with the same rule set as G' produces S with the highest probability?
- A simple Maximum Likelihood approach will do

$$\forall i : p(N_i \rightarrow \varphi_j) = \frac{|N_i \rightarrow \varphi_j|}{|N_i \rightarrow *|}$$

  - |.| Number of occurrence of a rule in the set of derivations
  - * Any rule consequence

Content of this Lecture

- Phrase-Structure Parse Trees
- Probabilistic Context-Free Grammars
- Parsing with PCFG
- Other Issues in Parsing

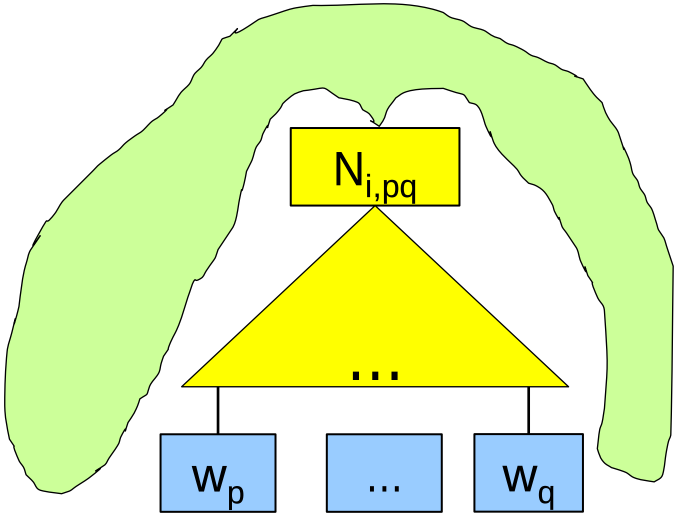
Issue 2: Evaluation

- Given a PCFG G and a sentence $s \in L(G)$: What is the overall probability of s given G?
  - We did not discuss this problem for HMM, but for PCFG it is simpler to derive parsing from evaluation
- Naïve: Find all derivations of s, sum-up their probabilities
  - Problem: There can be exponentially many derivations
- We give a Dynamic Programming based algorithm

Idea

- Recall that a PCFG build on a context-free grammar in CNF
- Definition
  The inside probability of a sub-sentence w_p ... w_q to be produced by a non-terminal N_i is defined as

$$\beta_i(p,q) = p(w_{pq} | N_{i,pq}, G)$$



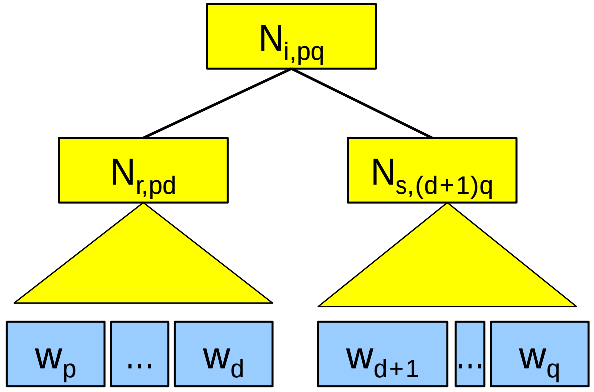
  - w_{pq}: Sub-sentence of s starting at token w_p at pos. p until token w_q at pos. q
  - $N_{i,pq}$: Non-terminal N_i producing w_{pq}
  - From now on, we omit the „G“ and “s”
- We search $\beta_S(1,n)$ for a sentence with n token

Induction

- We compute $\beta_S(1,n)$ by induction over the length of all sub-sentences
- Start: Assume p=q. Since we have a CNF, the rule producing w_{pp} must have the form $N_{i,pp} \rightarrow w_{pp}$.

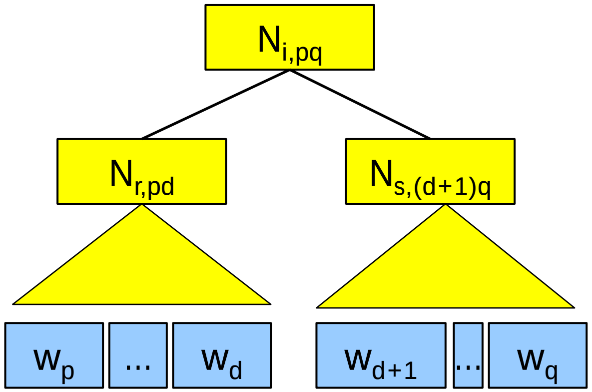
$$\beta_i(p,p) = p(w_{pp}|N_{i,pp}) = p(N_{i,pp} \rightarrow w_{pp})$$

  - This is parameter of G and can be lookup up for all (i,p)
- Induction: Assume p<q. Since we are in CNF, the derivation must look like this for some d with p≤d≤q



Derivation

- $\beta_i(p,q)$
  $= p(w_{pq}|N_{i,pq},G)$
  $= \ldots$



$$= \sum_{r,s} \sum_{d=p..q-1} p\left(w_{pd}, N_{r,pd}, w_{(d+1)q}, N_{s,(d+1)q} \mid N_{i,pq}\right)$$

$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_{r,pd}, N_{s,(d+1)q} \mid N_{i,pq}\right) * p\left(w_{pd} \mid N_{r,pd}, N_{s,(d+1)q}, N_{i,pq}\right) * $$
$$* \, p\left(w_{(d+1)q} \mid N_{r,pd}, N_{s,(d+1)q}, N_{i,pq}\right)$$

$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_{r,pd}, N_{s,(d+1)q} \mid N_{i,pq}\right) * p\left(w_{pd} \mid N_{r,pd}\right) * p\left(w_{(d+1)q} \mid N_{s,(d+1)q}\right)$$

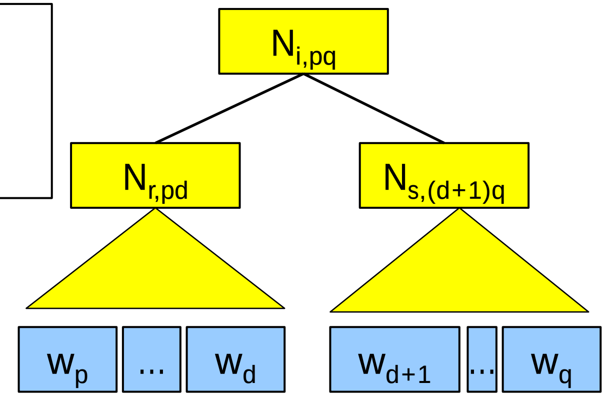
$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_i \rightarrow N_r \; N_s\right) * \beta_r(p,d) * \beta_s(d+1,q)$$

Derivation

All possible combinations of N_r/N_s producing their subtrees

Subtrees can have any length between 1 and q-p, sum is q-p

Chain rule of conditional probabilities



$$= \sum_{r,s} \sum_{d=p..q-1} p\left(w_{pd}, N_{r,pd}, w_{(d+1)q}, N_{s,(d+1)q} \mid N_{i,pq}\right)$$

$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_{r,pd}, N_{s,(d+1)q} \mid N_{i,pq}\right) * p\left(w_{pd} \mid N_{r,pd}, N_{s,(d+1)q}, N_{i,pq}\right) *$$

$$* \, p\left(w_{(d+1)q} \mid N_{r,pd}, N_{s,(d+1)q}, N_{i,pq}\right)$$

$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_{r,pd}, N_{s,(d+1)q} \mid N_{i,pq}\right) * p\left(w_{pd} \mid N_{r,pd}\right) * p\left(w_{(d+1)q} \mid N_{s,(d+1)q}\right)$$

$$= \sum_{r,s} \sum_{d=p..q-1} p\left(N_i \rightarrow N_r \; N_s\right) * \beta_r(p,d) * \beta_s(d+1,q)$$

Independence assumptions in CFG

Example

astronomer	saw	man	with	telescope

1: S → NP VP	1,00
2: PP → P NP	1,00
3: VP → V NP	0,70
4: VP → VP PP	0,30
5: P → with	1,00
6: V → saw	1,00

7: NP → NP PP	0,40
8: NP → astronomer	0,10
9: NP → telescope	0,18
10: NP → man	0,18
11: NP → saw	0,04
12: NP → ears	0,10

	1	2	3	4	5
1	$\beta_{NP}(1,1)=0,1$				
2		$\beta_{V}(2,2)=1$ $\beta_{NP}(2,2)=0,04$			
3			$\beta_{NP}(3,3)=0,18$		
4				$\beta_{P}(4,4)=1$	
5					$\beta_{NP}(5,5)=0,18$

Example

astronomer | saw | man | with | telescope

1: S → NP VP	1,00
2: PP → P NP	1,00
3: VP → V NP	0,70
4: VP → VP PP	0,30
5: P → with	1,00
6: V → saw	1,00

7: NP → NP PP	0,40
8: NP → astronomer	0,10
9: NP → telescope	0,18
10: NP → man	0,18
11: NP → saw	0,04
12: NP → ears	0,10

	1	2	3	4	5
1	β_{NP}=0,1	-			
2		β_V=1 β_{NP}=0,04	β_{VP}=0,7*1*0,18= 0,126		
3			β_{NP}=0,18	-	
4				β_P=1	β_{PP}=1*1*0,18= 0,18
5					β_{NP}=0,18

No rule X→NP V or X→NP NP

Must be VP→ V NP with p=0.7

Example

astronomer	saw	man	with	telescope

1: S → NP VP	1,00
2: PP → P NP	1,00
3: VP → V NP	0,70
4: VP → VP PP	0,30
5: P → with	1,00
6: V → saw	1,00

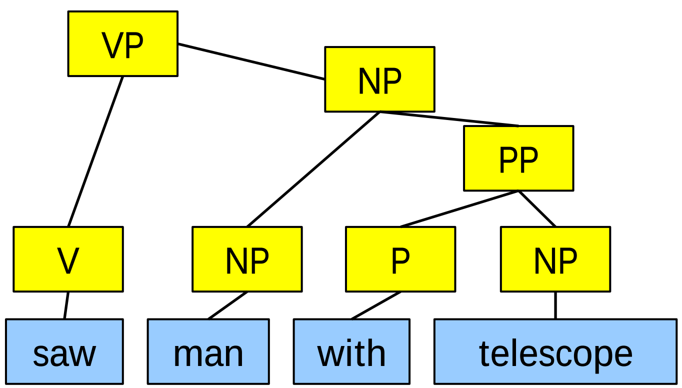
7: NP → NP PP	0,40
8: NP → astronomer	0,10
9: NP → telescope	0,18
10: NP → man	0,18
11: NP → saw	0,04
12: NP → ears	0,10

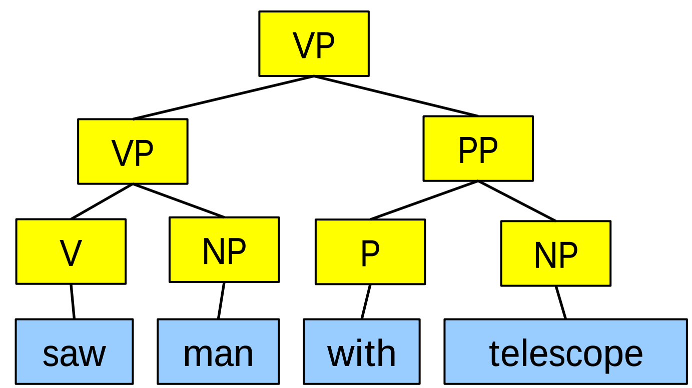
	1	2	3	4	5
1	β_{NP}=0,1	-	β_S=1*0,1*0,126= 0,0126		
2		β_V=1 β_{NP}=0,04	β_{VP}=0,126	-	
3			β_{NP}=0,18	-	β_{NP}=0,4*0,18*0,18= 0,1296
4				β_P=1	β_{PP}=0,18
5					β_{NP}=0,18

Example

astronomer | saw | man | with | telescope

	1	2	3	4	5
1	β_{NP}=0,1	-	β_{S}=0,0126	-	β_{VP}=0,0015…
2		β_{V}=1 β_{NP}=0,04	β_{VP}=0,126	-	$\beta_{VP1}+\beta_{VP2}$=0,015…
3			β_{NP}=0,18	-	β_{NP}=0,1296
4				β_{P}=1	β_{PP}=0,18
5					β_{NP}=0,18





Note

- This is the Cocke–Younger–Kasami (CYK) algorithm for parsing with context free grammars, enriched with aggregations / multiplications for computing probabilities
- Same complexity: $O(n^3*|G|)$
  - n: Sentence length
  - |G|: Number of rules in the grammar G

Note

- This is the Cocke–Younger–Kasami (CYK) algorithm for parsing with context free grammars, enriched with aggregations / multiplications for computing probabilities
- Same complexity: $O(n^3*|G|)$
  - n: Sentence length
  - |G|: Number of rules in the grammar G

Issue 1: Decoding / Parsing

- Once evaluation is solved, parsing is simple
- Instead of summing over all derivations, we only chose the most probable deviation of a sub-sentence for each possible root
- Let $\delta_i(p,q) = p(w_{pq}|N_{i,pq})$ be the most probable derivation of sub-sentence p..q from a non-terminal root N_i
- This gives

$$\delta_i(p,q) = \arg\max_{r,s}\left(\arg\max_{d=p\ldots q-1}\left(p\left(w_{pd}, N_{r,pd}, w_{(d+1)q}, N_{s,(d+1)q} \mid N_{i,pq}\right)\right)\right)$$

$$= \arg\max_{\substack{d=p\ldots q-1,\\ r,s}}\left(p(N_i \rightarrow N_r\ N_s) * \delta_r(p,d) * \delta_s(d+1,q)\right)$$

  – We omit induction start and backtracing

Content of this Lecture

- Phrase-Structure Parse Trees
- Probabilistic Context-Free Grammars
- Parsing with PCFG
- Other Issues in Parsing

Treebanks

- A treebank is a set of sentences (corpus) whose phrase structures are annotated
  - Training corpus for PCFG
  - Not many exist; very costly, manual task
- Most prominent: Penn Treebank
  - Marcus, Marcinkiewicz, Santorini. "Building a large annotated corpus of English: The Penn Treebank." Computational linguistics 19.2 (1993): 313-330.
    - ~5500 citations (!)
  - 2,499 stories from a 3-years Wall Street Journal (WSJ) collection
  - Roughly 1 Million tokens, freely available
- Deutsche Baumbanken
  - Deutsche Diachrone Baumbank, 3 historical periods, small
  - Tübinger Baumbank, 38.000 Sätze, 345.000 Token

Using Derivation History

- Phrase structure grammars as described here are kind-of simplistic
- One idea for improvement: Incorporate dependencies between non-terminals
  - Probability of rules is not identical across all positions in a sentence
  - Trick: Annotate derivation of a non-terminal in its name and learn different probabilities for different derivations

Read: NP generated from a VP → 7

1: S → NP VP
2: PP → P NP
3: VP → V NP
...
7: NP_{VP} → NP PP
7a: NP_{PP} → NP PP
...

Expansion	% as 1st Obj	% as 2nd Obj
NP → NNS	7.5%	0.2%
NP → PRP	13.4%	0.9%
NP → NP PP	12.2%	14.4%
NP → DT NN	10.4%	13.3%
NP → NNP	4.5%	5.9%
NP → NN	3.9%	9.2%
NP → JJ NN	1.1%	10.4%
NP → NP SBAR	0.3%	5.1%

Source: MS99; from Penn Treebank

Lexicalization

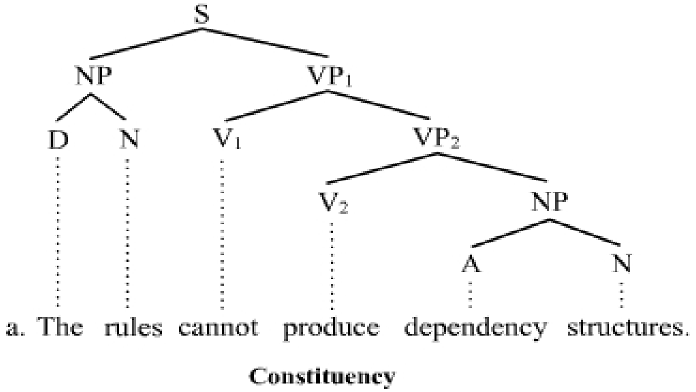
- Second idea: Incorporate word semantics (lexicalization)
  - Clearly, different verbs take different arguments leading to different structures (similar for other word types)
  - Trick: Learn a model for each head word of a non-terminal
    - VP_{walk}, VP_{take}, VP_{eat}, $VP_{...}$
  - Requires much larger training corpus and sophisticated smoothing

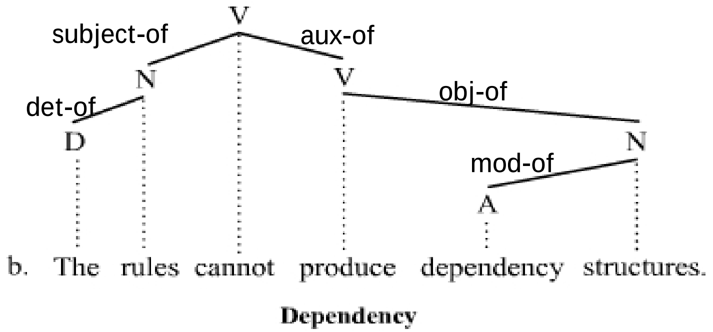
	Verb			
Local tree	*come*	*take*	*think*	*want*
V P - V	9.5%	2.6%	4.6%	5.7%
V P - V N P	1.1%	32.1%	0.2%	13.9%
VP → v PP	34.5%	3.1%	7.1%	0.3%
VP → V SBAR	6.6%	0.3%	73.0%	0.2%
VP → V S	2.2%	1.3%	4.8%	70.8%
VP → V NP S	0.1%	5.7%	0.0%	0.3%
VP → V PRT NP	0.3%	5.8%	0.0%	0.0%
VP → V PRT PP	6.1%	1.5%	0.2%	0.0%

Source: MS99; from Penn Treebank

Dependency Grammars

- Phrase structure grammars are not the only way to represent structural information within sentences
- Popular alternative: Dependency trees
  - Every word forms exactly one node
  - Edges describe the syntactic relationship between words: object-of, subject-of, modifier-of, preposition-of, ...
  - Different tag sets exist





Source: Wikipedia

Self-Assessment

- Which assumptions are behind PCFG for parsing?
- What is the complexity of the parsing problem in PCFG?
- Assume the following rule set ... Derive all derivations for the sentence ... together with their probabilities. Mark the most probable derivation.
- Derive the complexity of the decoding algorithm for PCFG
- What is the head word of a phrase in a phrase structure grammar?
- When are two grammars weakly equivalent?