



CFGs and PCFGs

(Probabilistic)
Context-Free
Grammars



A phrase structure grammar

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \text{people}$

$N \rightarrow \text{fish}$

$N \rightarrow \text{tanks}$

$N \rightarrow \text{rods}$

$V \rightarrow \text{people}$

$V \rightarrow \text{fish}$

$V \rightarrow \text{tanks}$

$P \rightarrow \text{with}$

people fish tanks

people fish with rods



Phrase structure grammars = context-free grammars (CFGs)

- $G = (T, N, S, R)$
 - T is a set of terminal symbols
 - N is a set of nonterminal symbols
 - S is the start symbol ($S \in N$)
 - R is a set of rules/productions of the form $X \rightarrow \gamma$
 - $X \in N$ and $\gamma \in (N \cup T)^*$
- A grammar G generates a language L .



Phrase structure grammars in NLP

- $G = (T, C, N, S, L, R)$
 - T is a set of terminal symbols
 - C is a set of preterminal symbols
 - N is a set of nonterminal symbols
 - S is the start symbol ($S \in N$)
 - L is the lexicon, a set of items of the form $X \rightarrow x$
 - $X \in P$ and $x \in T$
 - R is the grammar, a set of items of the form $X \rightarrow \gamma$
 - $X \in N$ and $\gamma \in (N \cup C)^*$
- By usual convention, S is the start symbol, but in statistical NLP, we usually have an extra node at the top (ROOT, TOP)
- We usually write e for an empty sequence, rather than nothing



A phrase structure grammar

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$

people fish tanks

people fish with rods



Probabilistic – or stochastic – context-free grammars (PCFGs)

- $G = (T, N, S, R, P)$
 - T is a set of terminal symbols
 - N is a set of nonterminal symbols
 - S is the start symbol ($S \in N$)
 - R is a set of rules/productions of the form $X \rightarrow \gamma$
 - P is a probability function
 - $P: R \rightarrow [0,1]$
 - $\forall X \in N, \sum_{X \rightarrow \gamma \in R} P(X \rightarrow \gamma) = 1$
- A grammar G generates a language model L .

$$\sum_{\gamma \in T^*} P(\gamma) = 1$$



A PCFG

$S \rightarrow NP VP$ 1.0

$VP \rightarrow V NP$ 0.6

$VP \rightarrow V NP PP$ 0.4

$NP \rightarrow NP NP$ 0.1

$NP \rightarrow NP PP$ 0.2

$NP \rightarrow N$ 0.7

$PP \rightarrow P NP$ 1.0

$N \rightarrow \textit{people}$ 0.5

$N \rightarrow \textit{fish}$ 0.2

$N \rightarrow \textit{tanks}$ 0.2

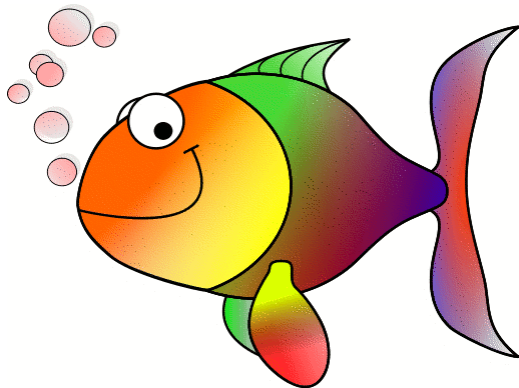
$N \rightarrow \textit{rods}$ 0.1

$V \rightarrow \textit{people}$ 0.1

$V \rightarrow \textit{fish}$ 0.6

$V \rightarrow \textit{tanks}$ 0.3

$P \rightarrow \textit{with}$ 1.0



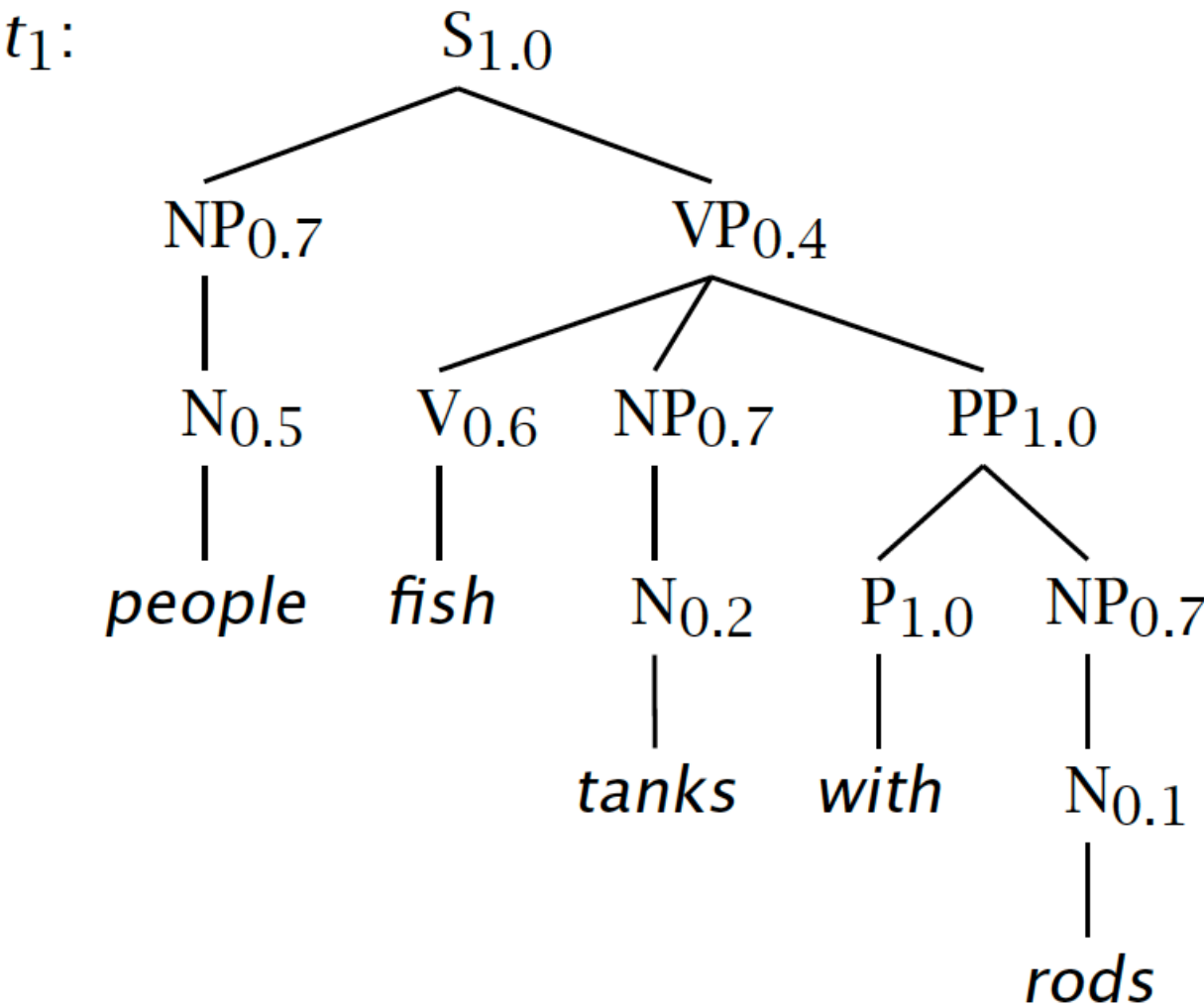
[With empty NP removed
so less ambiguous]



The probability of trees and strings

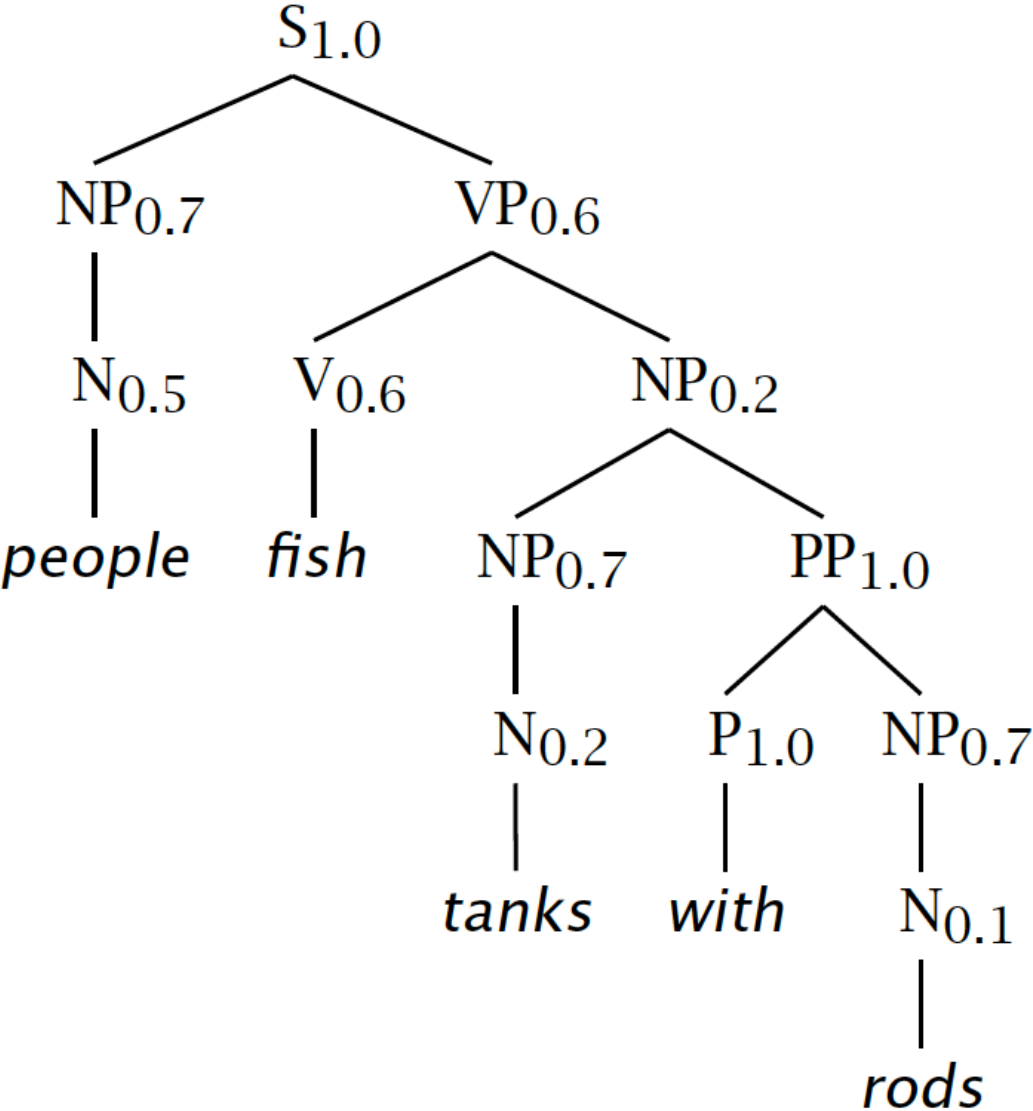
- $P(t)$ – The probability of a tree t is the product of the probabilities of the rules used to generate it.
- $P(s)$ – The probability of the string s is the sum of the probabilities of the trees which have that string as their yield

$$\begin{aligned} P(s) &= \sum_j P(s, t) \text{ where } t \text{ is a parse of } s \\ &= \sum_j P(t) \end{aligned}$$





t_2 :



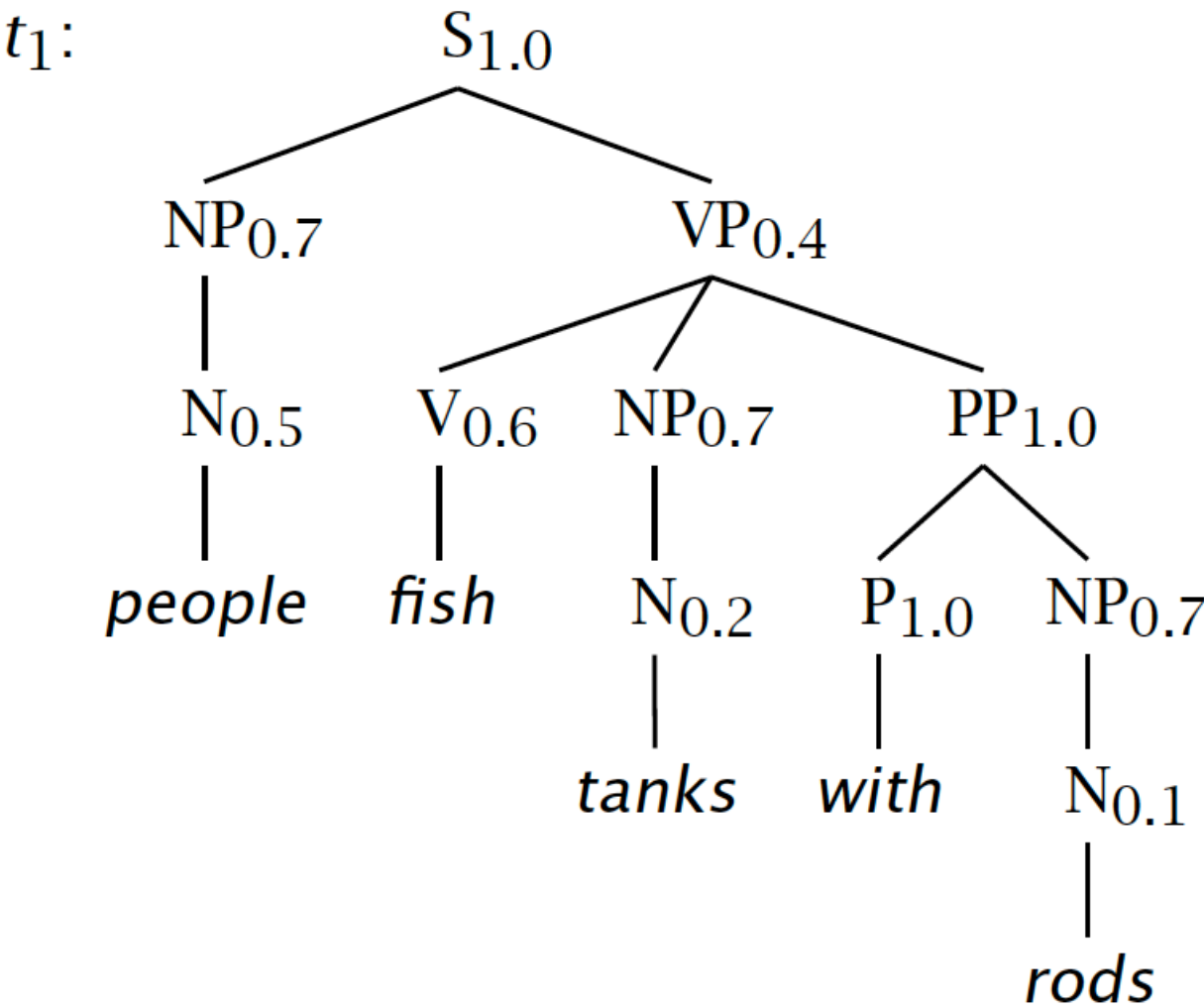


Tree and String Probabilities

- $s = \textit{people fish tanks with rods}$
- $P(t_1) = 1.0 \times 0.7 \times 0.4 \times 0.5 \times 0.6 \times 0.7$
 $\times 1.0 \times 0.2 \times 1.0 \times 0.7 \times 0.1$
 $= 0.0008232$
- $P(t_2) = 1.0 \times 0.7 \times 0.6 \times 0.5 \times 0.6 \times 0.2$
 $\times 0.7 \times 1.0 \times 0.2 \times 1.0 \times 0.7 \times 0.1$
 $= 0.00024696$
- $P(s) = P(t_1) + P(t_2)$
 $= 0.0008232 + 0.00024696$
 $= 0.00107016$

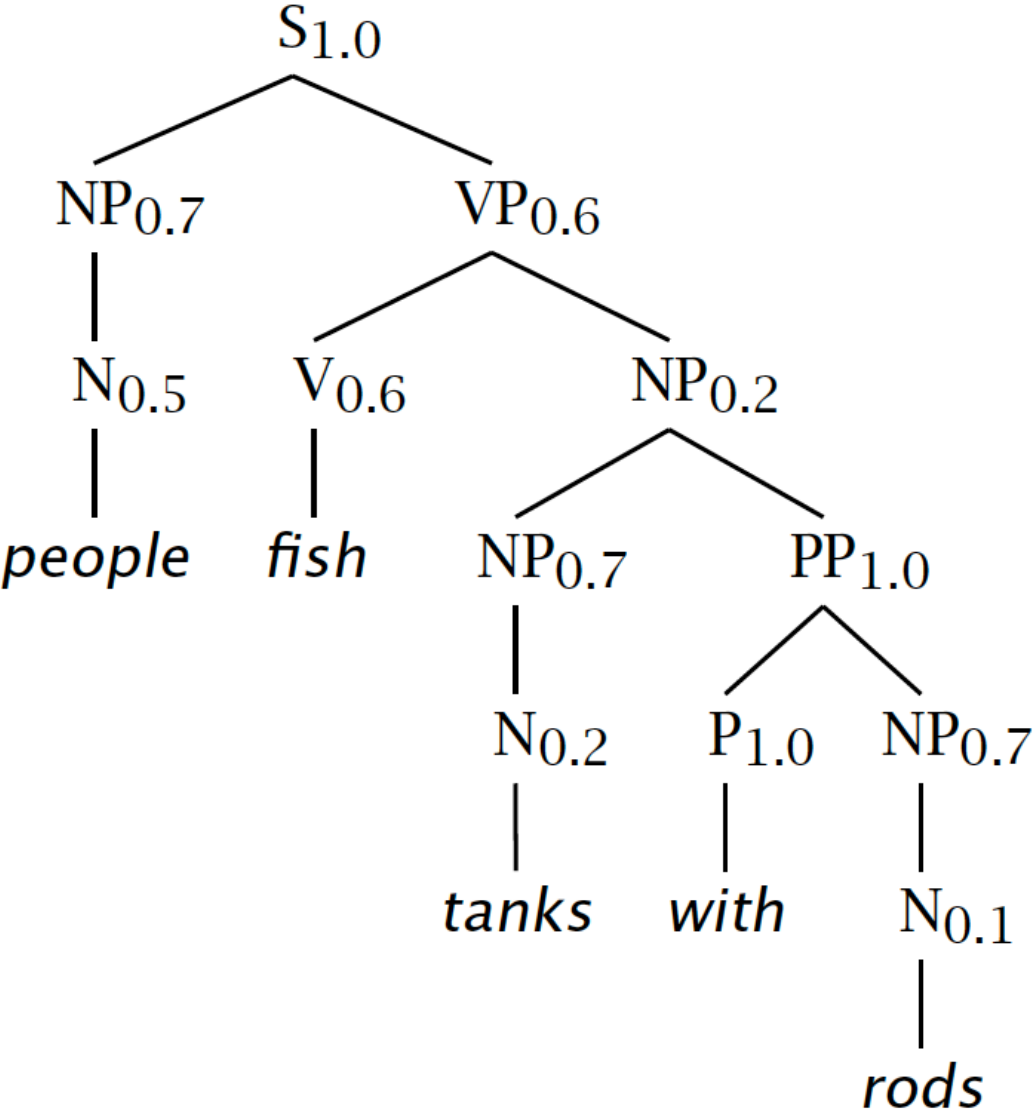
Verb attach

Noun attach





t_2 :





CFGs and PCFGs

(Probabilistic) Context-Free Grammars

Restricting the grammar form for efficient parsing



Chomsky Normal Form

- All rules are of the form $X \rightarrow YZ$ or $X \rightarrow w$
 - $X, Y, Z \in N$ and $w \in T$
- A transformation to this form doesn't change the weak generative capacity of a CFG
 - That is, it recognizes the same language
 - But maybe with different trees
- Empties and unaries are removed recursively
- n -ary rules are divided by introducing new nonterminals ($n > 2$)



A phrase structure grammar

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$S \rightarrow VP$

$VP \rightarrow V NP$

$VP \rightarrow V$

$VP \rightarrow V NP PP$

$VP \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP$

$NP \rightarrow NP PP$

$NP \rightarrow PP$

$NP \rightarrow N$

$PP \rightarrow P NP$

$PP \rightarrow P$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V$

$S \rightarrow V$

$VP \rightarrow V NP PP$

$S \rightarrow V NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP$

$NP \rightarrow NP PP$

$NP \rightarrow PP$

$NP \rightarrow N$

$PP \rightarrow P NP$

$PP \rightarrow P$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V$

$VP \rightarrow V NP PP$

$S \rightarrow V NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP$

$NP \rightarrow NP PP$

$NP \rightarrow PP$

$NP \rightarrow N$

$PP \rightarrow P NP$

$PP \rightarrow P$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$S \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$S \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$S \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V NP PP$

$S \rightarrow V NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP$

$NP \rightarrow NP PP$

$NP \rightarrow PP$

$NP \rightarrow N$

$PP \rightarrow P NP$

$PP \rightarrow P$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$S \rightarrow \textit{people}$

$VP \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$S \rightarrow \textit{fish}$

$VP \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$S \rightarrow \textit{tanks}$

$VP \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V NP PP$

$S \rightarrow V NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow P NP$

$PP \rightarrow P NP$

$NP \rightarrow \textit{people}$

$NP \rightarrow \textit{fish}$

$NP \rightarrow \textit{tanks}$

$NP \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$S \rightarrow \textit{people}$

$VP \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$S \rightarrow \textit{fish}$

$VP \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$S \rightarrow \textit{tanks}$

$VP \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$

$PP \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V @VP_V$

$@VP_V \rightarrow NP PP$

$S \rightarrow V @S_V$

$@S_V \rightarrow NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow P NP$

$PP \rightarrow P NP$

$NP \rightarrow \textit{people}$

$NP \rightarrow \textit{fish}$

$NP \rightarrow \textit{tanks}$

$NP \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$S \rightarrow \textit{people}$

$VP \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$S \rightarrow \textit{fish}$

$VP \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$S \rightarrow \textit{tanks}$

$VP \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$

$PP \rightarrow \textit{with}$



A phrase structure grammar

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \textit{people}$

$N \rightarrow \textit{fish}$

$N \rightarrow \textit{tanks}$

$N \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$



Chomsky Normal Form steps

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$S \rightarrow V NP$

$VP \rightarrow V @VP_V$

$@VP_V \rightarrow NP PP$

$S \rightarrow V @S_V$

$@S_V \rightarrow NP PP$

$VP \rightarrow V PP$

$S \rightarrow V PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow P NP$

$PP \rightarrow P NP$

$NP \rightarrow \textit{people}$

$NP \rightarrow \textit{fish}$

$NP \rightarrow \textit{tanks}$

$NP \rightarrow \textit{rods}$

$V \rightarrow \textit{people}$

$S \rightarrow \textit{people}$

$VP \rightarrow \textit{people}$

$V \rightarrow \textit{fish}$

$S \rightarrow \textit{fish}$

$VP \rightarrow \textit{fish}$

$V \rightarrow \textit{tanks}$

$S \rightarrow \textit{tanks}$

$VP \rightarrow \textit{tanks}$

$P \rightarrow \textit{with}$

$PP \rightarrow \textit{with}$

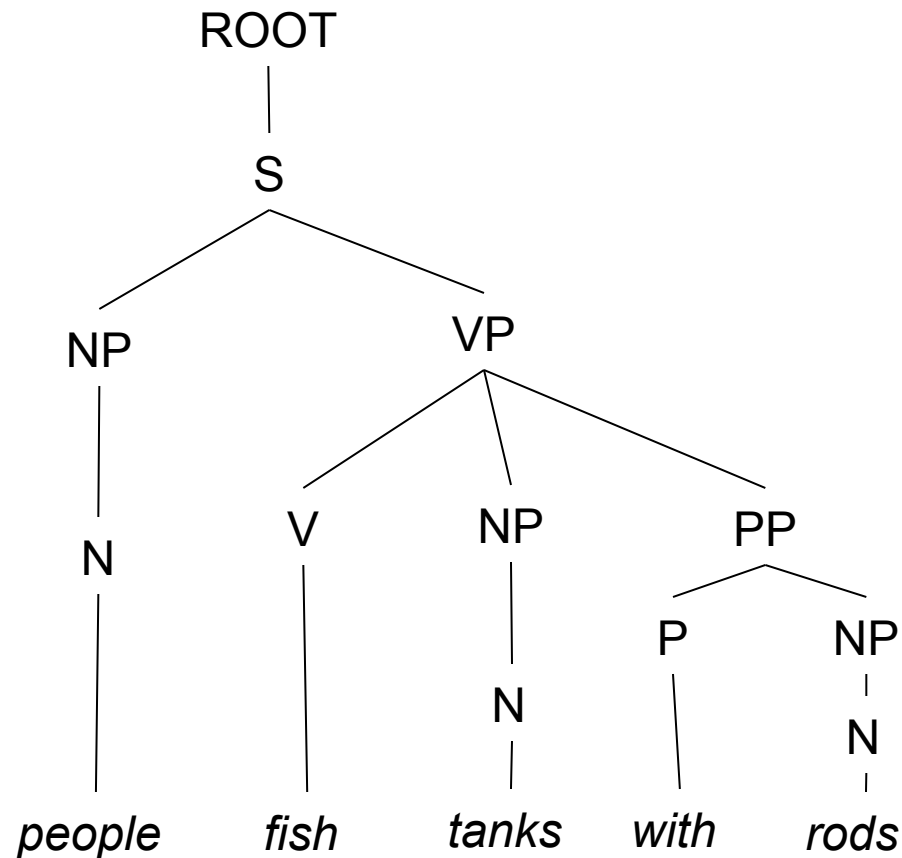


Chomsky Normal Form

- You should think of this as a transformation for efficient parsing
- With some extra book-keeping in symbol names, you can even reconstruct the same trees with a detransform
- In practice full Chomsky Normal Form is a pain
 - Reconstructing n-aries is easy
 - Reconstructing unaries/empties is trickier
- **Binarization** is crucial for cubic time CFG parsing
- The rest isn't necessary; it just makes the algorithms cleaner and a bit quicker

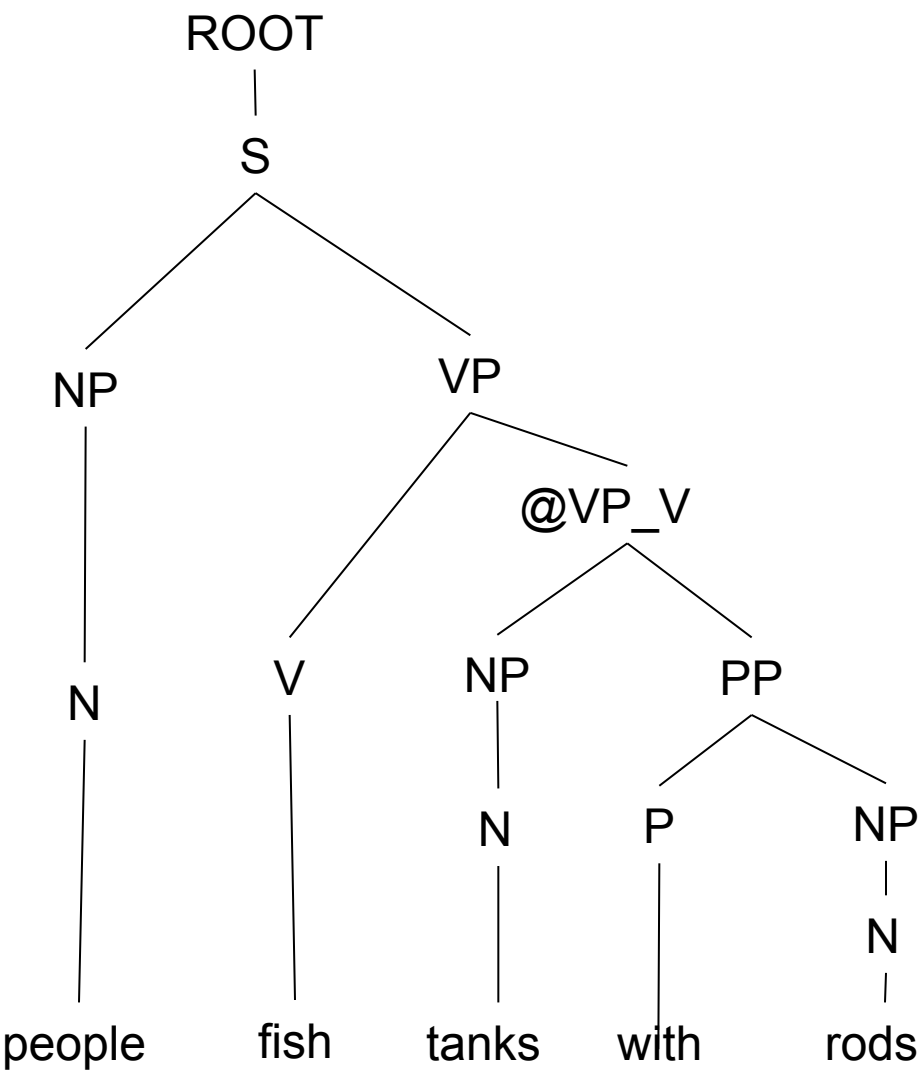


An example: before binarization...



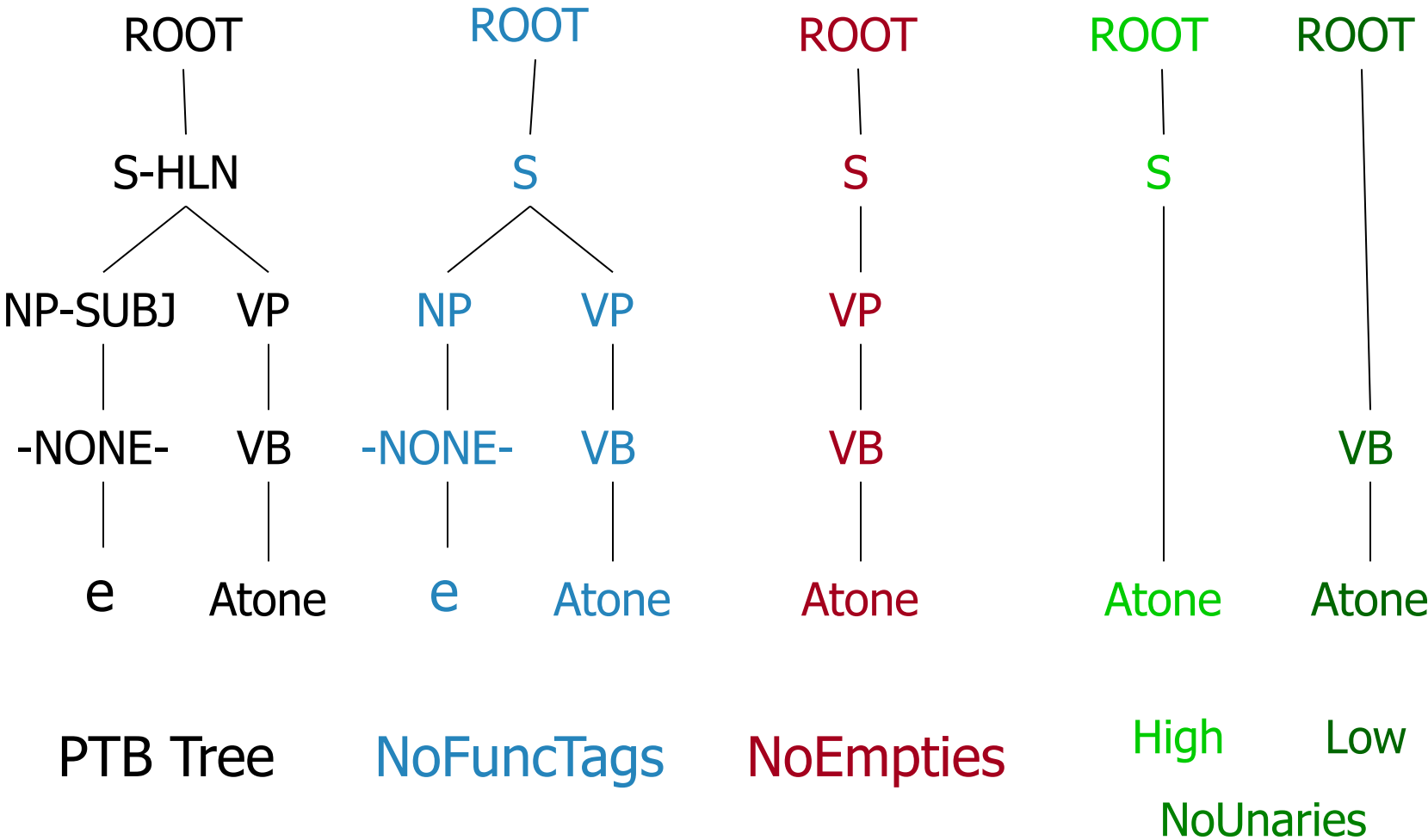


After binarization...

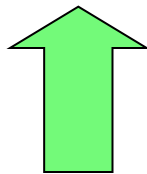




Treebank: empties and unaries

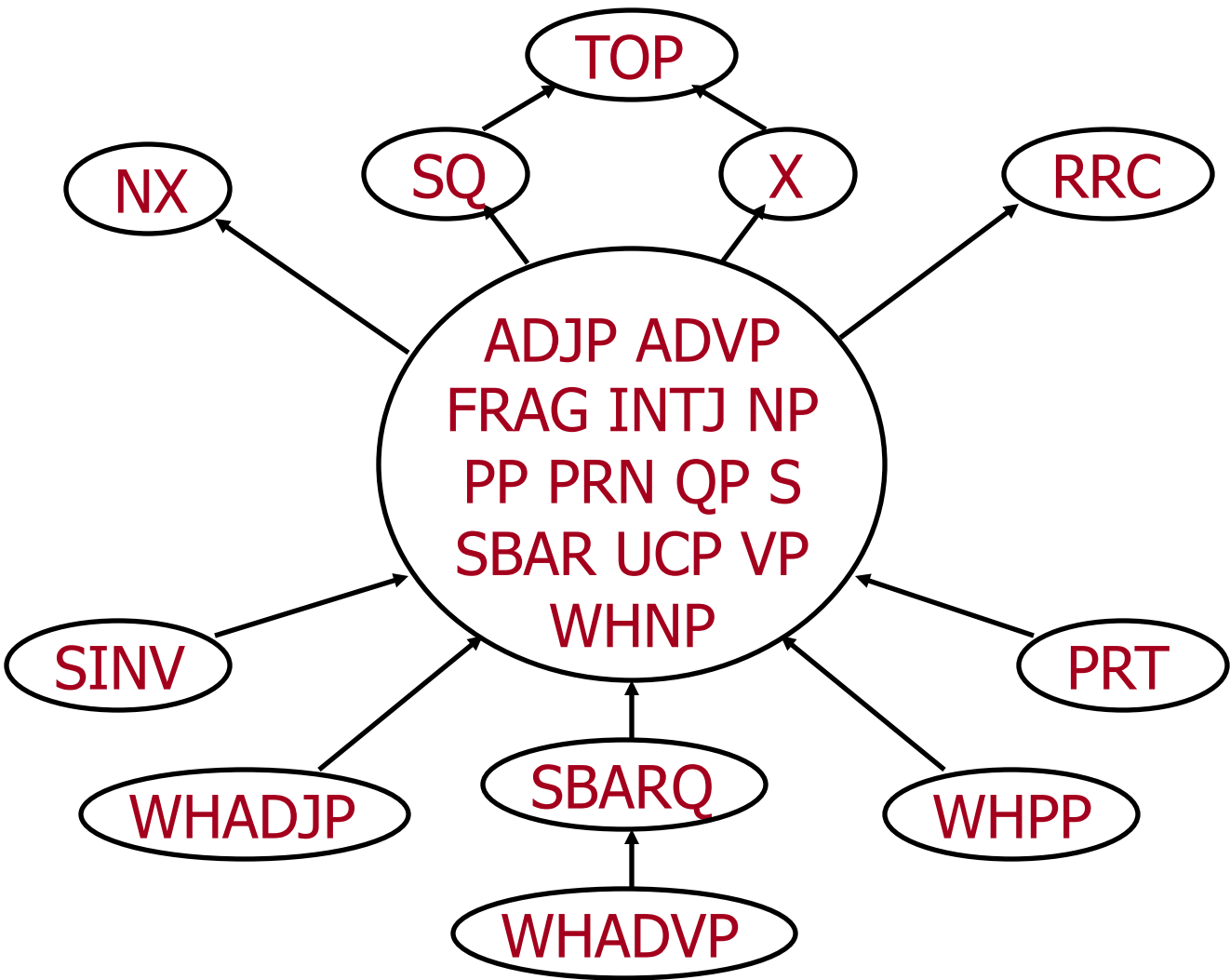


Unary rules: alchemy in the land of treebanks





Same-Span Reachability



NoEmpties

- LST
- CONJP
- NAC

Restricting the grammar form for efficient parsing

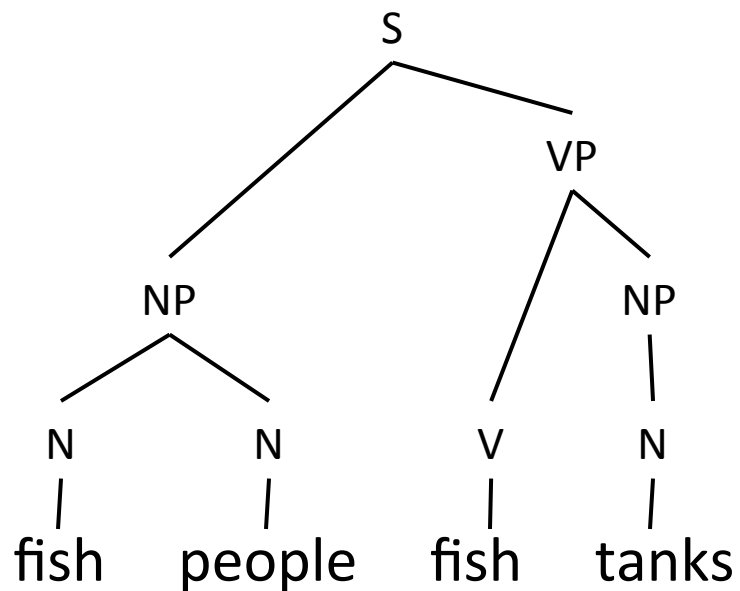


CKY Parsing

Exact polynomial time parsing of (P)CFGs



Constituency Parsing



PCFG

Rule Prob θ_i

$S \rightarrow NP VP$ θ_0

$NP \rightarrow NP NP$ θ_1

...

$N \rightarrow \text{fish}$ θ_{42}

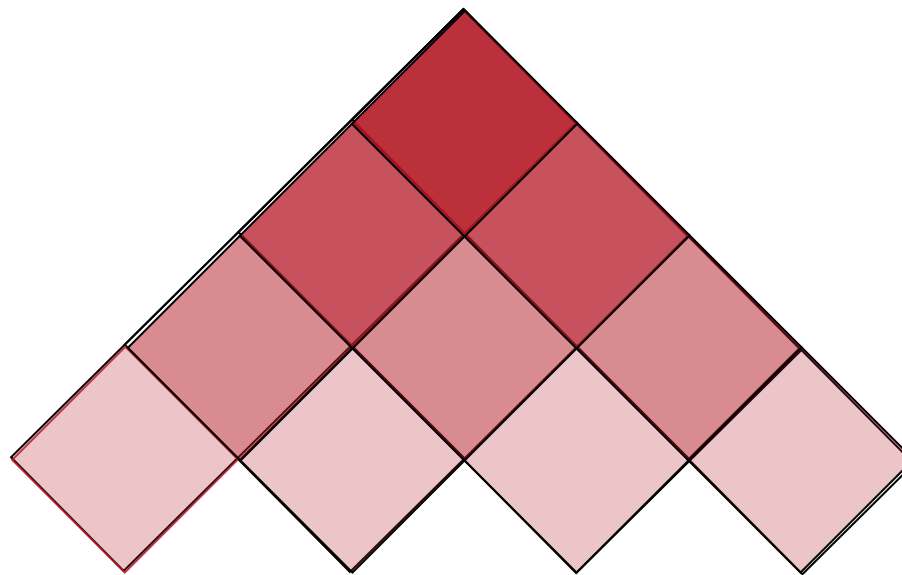
$N \rightarrow \text{people}$ θ_{43}

$V \rightarrow \text{fish}$ θ_{44}

...



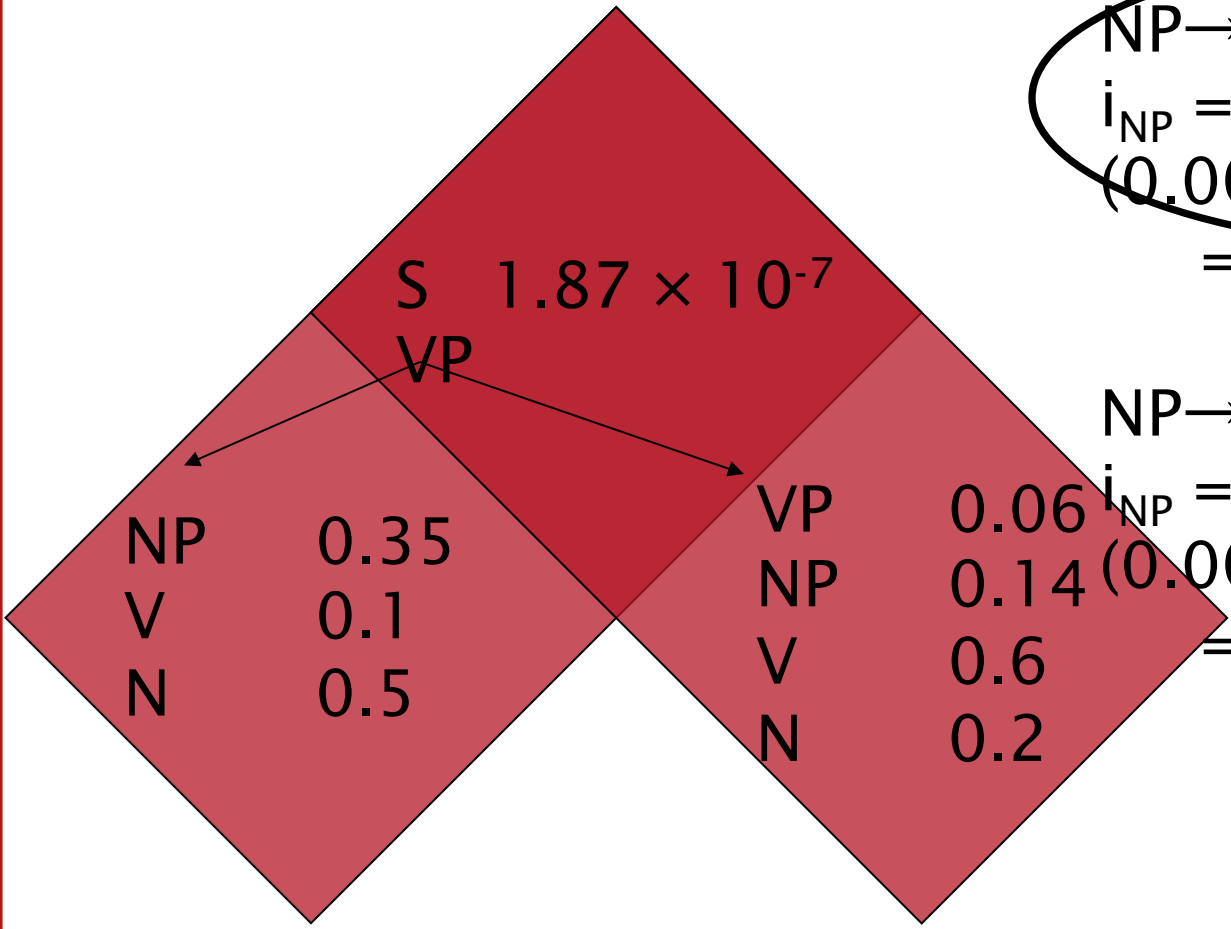
Cocke-Kasami-Younger (CKY) Constituency Parsing



fish people fish tanks



Viterbi (Max) Scores



NP→NN NNS 0.13
 $i_{NP} = (0.13)(0.0023)$
 (0.0014)
 $= 1.87 \times 10^{-7}$

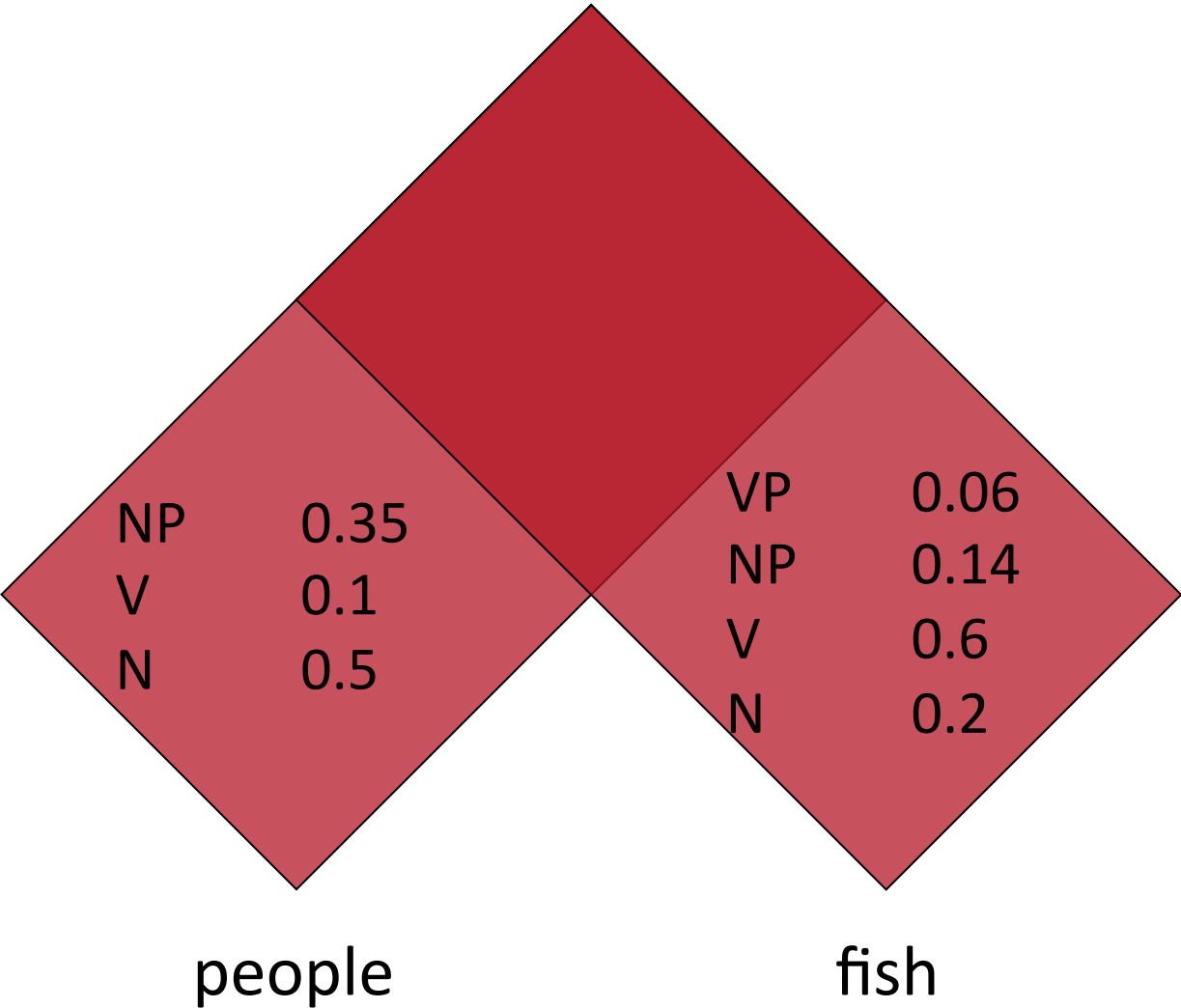
NP→NNP NNS 0.056
 $i_{NP} = (0.056)(0.001)$
 (0.0014)
 $= 7.84 \times 10^{-8}$

people

fish



Viterbi (Max) Scores



$S \rightarrow NP VP$	0.9
$S \rightarrow VP$	0.1
$VP \rightarrow V NP$	0.5
$VP \rightarrow V$	0.1
$VP \rightarrow V @VP_V$	0.3
$VP \rightarrow V PP$	0.1
$@VP_V \rightarrow NP PP$	1.0
$NP \rightarrow NP NP$	0.1
$NP \rightarrow NP PP$	0.2
$NP \rightarrow N$	0.7
$PP \rightarrow P NP$	1.0



Extended CKY parsing

- Unaries can be incorporated into the algorithm
 - Messy, but doesn't increase algorithmic complexity
- Empties can be incorporated
 - Use fenceposts
 - Doesn't increase complexity; essentially like unaries
- Binarization is *vital*
 - Without binarization, you don't get parsing cubic in the length of the sentence and in the number of nonterminals in the grammar
 - Binarization may be an explicit transformation or implicit in how the parser works (Early-style dotted rules), but it's always there.



The CKY algorithm (1960/1965) ... extended to unaries

```

function CKY(words, grammar) returns [most_probable_parse, prob]
  score = new double[#(words)+1][#(words)+1][#(nonterms)]
  back = new Pair[#(words)+1][#(words)+1][#(nonterms)]
  for i=0; i<#(words); i++
    for A in nonterms
      if A -> words[i] in grammar
        score[i][i+1][A] = P(A -> words[i])
  //handle unaries
  boolean added = true
  while added
    added = false
    for A, B in nonterms
      if score[i][i+1][B] > 0 && A->B in grammar
        prob = P(A->B)*score[i][i+1][B]
        if prob > score[i][i+1][A]
          score[i][i+1][A] = prob
          back[i][i+1][A] = B
          added = true

```



The CKY algorithm (1960/1965)

... extended to unaries

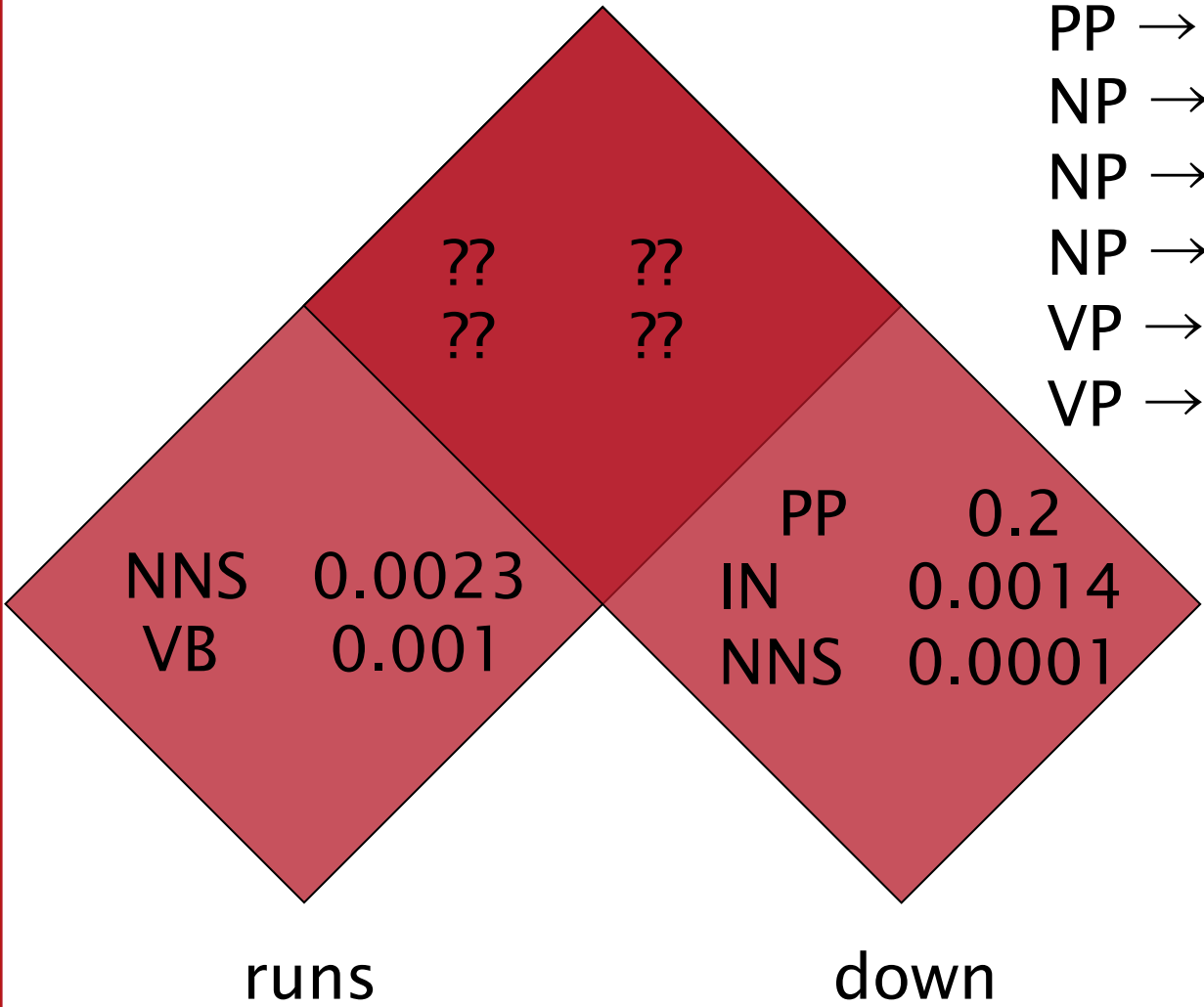
```

for span = 2 to #(words)
  for begin = 0 to #(words)- span
    end = begin + span
    for split = begin+1 to end-1
      for A,B,C in nonterms
        prob=score[begin][split][B]*score[split][end][C]*P(A->BC)
        if prob > score[begin][end][A]
          score[begin][end][A] = prob
          back[begin][end][A] = new Triple(split,B,C)
      //handle unaries
      boolean added = true
      while added
        added = false
        for A, B in nonterms
          prob = P(A->B)*score[begin][end][B];
          if prob > score[begin][end][A]
            score[begin][end][A] = prob
            back[begin][end][A] = B
            added = true
    return buildTree(score, back)

```




Quiz Question!



PP → IN	0.002
NP → NNS NNS	0.01
NP → NNS NP	0.005
NP → NNS PP	0.01
VP → VB PP	0.045
VP → VB NP	0.015

What constituents (with what probability can you make?



CKY Parsing

Exact polynomial time parsing of (P)CFGs



CKY Parsing

A worked example



The grammar: Binary, no epsilons,

$S \rightarrow NP VP$	0.9	$N \rightarrow people$	0.5
$S \rightarrow VP$	0.1	$N \rightarrow fish$	0.2
$VP \rightarrow V NP$	0.5	$N \rightarrow tanks$	0.2
$VP \rightarrow V$	0.1	$N \rightarrow rods$	0.1
$VP \rightarrow V @VP_V$	0.3	$V \rightarrow people$	0.1
$VP \rightarrow V PP$	0.1	$V \rightarrow fish$	0.6
$@VP_V \rightarrow NP PP$	1.0	$V \rightarrow tanks$	0.3
$NP \rightarrow NP NP$	0.1	$P \rightarrow with$	1.0
$NP \rightarrow NP PP$	0.2		
$NP \rightarrow N$	0.7		
$PP \rightarrow P NP$	1.0		

	fish	1	people	2	fish	3	tanks	4
0	score[0][1]	score[0][2]	score[0][3]	score[0][4]				
1		score[1][2]	score[1][3]	score[1][4]				
2			score[2][3]	score[2][4]				
3				score[3][4]				
4								

$S \rightarrow NP VP$	0.9	0	fish	1	people	2	fish	3	tanks	4
$S \rightarrow VP$	0.1									
$VP \rightarrow V NP$	0.5									
$VP \rightarrow V$	0.1									
$VP \rightarrow V @VP_V$	0.3	1								
$VP \rightarrow V PP$	0.1									
$@VP_V \rightarrow NP PP$	1.0									
$NP \rightarrow NP NP$	0.1									
$NP \rightarrow NP PP$	0.2									
$NP \rightarrow N$	0.7	2								
$PP \rightarrow P NP$	1.0									
$N \rightarrow people$	0.5									
$N \rightarrow fish$	0.2									
$N \rightarrow tanks$	0.2									
$N \rightarrow rods$	0.1									
$V \rightarrow people$	0.1									
$V \rightarrow fish$	0.6									
$V \rightarrow tanks$	0.3									
$P \rightarrow with$	1.0									

3

for i=0; i<#(words); i++
 for A in nonterms
 if A -> words[i] in grammar
 score[i][i+1][A] = P(A -> words[i]);

4

$S \rightarrow NP VP$	0.9
$S \rightarrow VP$	0.1
$VP \rightarrow V NP$	0.5
$VP \rightarrow V$	0.1
$VP \rightarrow V @VP_V$	0.3
$VP \rightarrow V PP$	0.1
$@VP_V \rightarrow NP PP$	1.0
$NP \rightarrow NP NP$	0.1
$NP \rightarrow NP PP$	0.2
$NP \rightarrow N$	0.7
$PP \rightarrow P NP$	1.0
$N \rightarrow people$	0.5
$N \rightarrow fish$	0.2
$N \rightarrow tanks$	0.2
$N \rightarrow rods$	0.1
$V \rightarrow people$	0.1
$V \rightarrow fish$	0.6
$V \rightarrow tanks$	0.3
$P \rightarrow with$	1.0

	fish	1	people	2	fish	3	tanks	4
0	N \rightarrow fish 0.2 V \rightarrow fish 0.6							
1			N \rightarrow people 0.5 V \rightarrow people 0.1					
2				N \rightarrow fish 0.2 V \rightarrow fish 0.6				
							N \rightarrow tanks 0.2 V \rightarrow tanks 0.1	

```
// handle unaries
boolean added = true
while added
  added = false
  for A, B in nonterms
    if score[i][i+1][B] > 0 && A->B in grammar
      prob = P(A->B)*score[i][i+1][B]
      if(prob > score[i][i+1][A])
        score[i][i+1][A] = prob
        back[i][i+1][A] = B
        added = true
```

		fish	1	people	2	fish	3	tanks	4
$S \rightarrow NP VP$	0.9	0	1						
$S \rightarrow VP$	0.1								
$VP \rightarrow V NP$	0.5								
$VP \rightarrow V$	0.1								
$VP \rightarrow V @VP_V$	0.3								
$VP \rightarrow V PP$	0.1	2		N \rightarrow people 0.5 V \rightarrow people 0.1 NP \rightarrow N 0.35 VP \rightarrow V 0.01 S \rightarrow VP 0.001					
@VP_V \rightarrow NP PP	1.0								
NP \rightarrow NP NP	0.1								
NP \rightarrow NP PP	0.2								
NP \rightarrow N	0.7								
PP \rightarrow P NP	1.0	3			N \rightarrow fish 0.2 V \rightarrow fish 0.6 NP \rightarrow N 0.14 VP \rightarrow V 0.06 S \rightarrow VP 0.006				
N \rightarrow people	0.5								
N \rightarrow fish	0.2								
N \rightarrow tanks	0.2								
N \rightarrow rods	0.1								
V \rightarrow people	0.1	4					N \rightarrow tanks 0.2 V \rightarrow tanks 0.1 NP \rightarrow N 0.14 VP \rightarrow V 0.03 S \rightarrow VP 0.003		
V \rightarrow fish	0.6								
V \rightarrow tanks	0.3								
P \rightarrow with	1.0								

- $S \rightarrow NP VP$ 0.9
- $S \rightarrow VP$ 0.1
- $VP \rightarrow V NP$ 0.5
- $VP \rightarrow V$ 0.1
- $VP \rightarrow V @VP_V$ 0.3
- $VP \rightarrow V PP$ 0.1
- $@VP_V \rightarrow NP PP$ 1.0
- $NP \rightarrow NP NP$ 0.1
- $NP \rightarrow NP PP$ 0.2
- $NP \rightarrow N$ 0.7
- $PP \rightarrow P NP$ 1.0
- $N \rightarrow people$ 0.5
- $N \rightarrow fish$ 0.2
- $N \rightarrow tanks$ 0.2
- $N \rightarrow rods$ 0.1
- $V \rightarrow people$ 0.1
- $V \rightarrow fish$ 0.6
- $V \rightarrow tanks$ 0.3
- $P \rightarrow with$ 1.0

	fish	1	people	2	fish	3	tanks	4
0								
1	<div>N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006</div>		<div>NP → NP NP 0.0049 VP → V NP 0.105 S → NP VP 0.00126</div>					
2			<div>N → people 0.5 V → people 0.1 NP → N 0.35 VP → V 0.01 S → VP 0.001</div>	<div>NP → NP NP 0.0049 VP → V NP 0.007 S → NP VP 0.0189</div>				
3				<div>N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006</div>		<div>NP → NP NP 0.00196 VP → V NP 0.042 S → NP VP 0.00378</div>		
4						<div>N → tanks 0.2 V → tanks 0.1 NP → N 0.14 VP → V 0.03 S → VP 0.003</div>		

//handle unaries
boolean added = true
while added
 added = false
 for A, B in nonterms
 prob = P(A->B)*score[begin][end][B];
 if prob > score[begin][end][A]
 score[begin][end][A] = prob
 back[begin][end][A] = B
 added = true

```
//handle unaries
boolean added = true
while added
    added = false
    for A, B in nonterms
        prob = P(A->B)*score[begin][end][B];
        if prob > score[begin][end][A]
            score[begin][end][A] = prob
            back[begin][end][A] = B
    added = true
```

$S \rightarrow NP VP$	0.9
$S \rightarrow VP$	0.1
$VP \rightarrow V NP$	0.5
$VP \rightarrow V$	0.1
$VP \rightarrow V @VP_V$	0.3
$VP \rightarrow V PP$	0.1
$@VP_V \rightarrow NP PP$	1.0
$NP \rightarrow NP NP$	0.1
$NP \rightarrow NP PP$	0.2
$NP \rightarrow N$	0.7
$PP \rightarrow P NP$	1.0
$N \rightarrow people$	0.5
$N \rightarrow fish$	0.2
$N \rightarrow tanks$	0.2
$N \rightarrow rods$	0.1
$V \rightarrow people$	0.1
$V \rightarrow fish$	0.6
$V \rightarrow tanks$	0.3
$P \rightarrow with$	1.0

	fish	1	people	2	fish	3	tanks	4	
0									
1	N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006	NP → NP NP 0.0049 VP → V NP 0.105 S → VP 0.0105							
2		N → people 0.5 V → people 0.1 NP → N 0.35 VP → V 0.01 S → VP 0.001		NP → NP NP 0.0049 VP → V NP 0.007 S → NP VP 0.0189					
3				N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006		NP → NP NP 0.00196 VP → V NP 0.042 S → VP 0.0042			
4	<div>for split = begin+1 to end-1 for A,B,C in nonterms prob=score[begin][split][B]*score[split][end][C]*P(A->BC) if prob > score[begin][end][A] score[begin][end][A] = prob back[begin][end][A] = new Triple(split,B,C)</div>					N → tanks 0.2 V → tanks 0.1 NP → N 0.14 VP → V 0.03 S → VP 0.003			

```

for split = begin+1 to end-1
  for A,B,C in nonterms
    prob=score[begin][split][B]*score[split][end][C]*P(A->BC)
    if prob > score[begin][end][A]
      score[begin][end][A] = prob
      back[begin][end][A] = new Triple(split,B,C)

```

- $S \rightarrow NP VP$ 0.9
- $S \rightarrow VP$ 0.1
- $VP \rightarrow V NP$ 0.5
- $VP \rightarrow V$ 0.1
- $VP \rightarrow V @VP_V$ 0.3
- $VP \rightarrow V PP$ 0.1
- $@VP_V \rightarrow NP PP$ 1.0
- $NP \rightarrow NP NP$ 0.1
- $NP \rightarrow NP PP$ 0.2
- $NP \rightarrow N$ 0.7
- $PP \rightarrow P NP$ 1.0
- $N \rightarrow people$ 0.5
- $N \rightarrow fish$ 0.2
- $N \rightarrow tanks$ 0.2
- $N \rightarrow rods$ 0.1
- $V \rightarrow people$ 0.1
- $V \rightarrow fish$ 0.6
- $V \rightarrow tanks$ 0.3
- $P \rightarrow with$ 1.0

	fish	1	people	2	fish	3	tanks	4
0								
1	N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006	NP → NP NP 0.0049 VP → V NP 0.105 S → VP 0.0105	NP → NP NP 0.0000686 VP → V NP 0.00147 S → NP VP 0.000882					
		N → people 0.5 V → people 0.1 NP → N 0.35 VP → V 0.01 S → VP 0.001	NP → NP NP 0.0049 VP → V NP 0.007 S → NP VP 0.0189					
2				N → fish 0.2 V → fish 0.6 NP → N 0.14 VP → V 0.06 S → VP 0.006	NP → NP NP 0.00196 VP → V NP 0.042 S → VP 0.0042			
3					N → tanks 0.2 V → tanks 0.1 NP → N 0.14 VP → V 0.03 S → VP 0.003			
4	<div>for split = begin+1 to end-1 for A,B,C in nonterms prob=score[begin][split][B]*score[split][end][C]*P(A->BC) if prob > score[begin][end][A] score[begin][end][A] = prob back[begin][end][A] = new Triple(split,B,C)</div>							

$S \rightarrow NP VP$	0.9
$S \rightarrow VP$	0.1
$VP \rightarrow V NP$	0.5
$VP \rightarrow V$	0.1
$VP \rightarrow V @VP_V$	0.3
$VP \rightarrow V PP$	0.1
$@VP_V \rightarrow NP PP$	1.0
$NP \rightarrow NP NP$	0.1
$NP \rightarrow NP PP$	0.2
$NP \rightarrow N$	0.7
$PP \rightarrow P NP$	1.0
$N \rightarrow people$	0.5
$N \rightarrow fish$	0.2
$N \rightarrow tanks$	0.2
$N \rightarrow rods$	0.1
$V \rightarrow people$	0.1
$V \rightarrow fish$	0.6
$V \rightarrow tanks$	0.3
$P \rightarrow with$	1.0

	fish	1	people	2	fish	3	tanks	4
0								
1	$N \rightarrow fish\ 0.2$ $V \rightarrow fish\ 0.6$ $NP \rightarrow N\ 0.14$ $VP \rightarrow V\ 0.06$ $S \rightarrow VP\ 0.006$		$NP \rightarrow NP\ NP$ 0.0049 $VP \rightarrow V\ NP$ 0.105 $S \rightarrow VP$ 0.0105		$NP \rightarrow NP\ NP$ 0.0000686 $VP \rightarrow V\ NP$ 0.00147 $S \rightarrow NP\ VP$ 0.000882			
2			$N \rightarrow people\ 0.5$ $V \rightarrow people\ 0.1$ $NP \rightarrow N\ 0.35$ $VP \rightarrow V\ 0.01$ $S \rightarrow VP\ 0.001$		$NP \rightarrow NP\ NP$ 0.0049 $VP \rightarrow V\ NP$ 0.007 $S \rightarrow NP\ VP$ 0.0189		$NP \rightarrow NP\ NP$ 0.0000686 $VP \rightarrow V\ NP$ 0.000098 $S \rightarrow NP\ VP$ 0.01323	
3					$N \rightarrow fish\ 0.2$ $V \rightarrow fish\ 0.6$ $NP \rightarrow N\ 0.14$ $VP \rightarrow V\ 0.06$ $S \rightarrow VP\ 0.006$		$NP \rightarrow NP\ NP$ 0.00196 $VP \rightarrow V\ NP$ 0.042 $S \rightarrow VP$ 0.0042	
4							$N \rightarrow tanks\ 0.2$ $V \rightarrow tanks\ 0.1$ $NP \rightarrow N\ 0.14$ $VP \rightarrow V\ 0.03$ $S \rightarrow VP\ 0.003$	

```

for split = begin+1 to end-1
  for A,B,C in nonterms
    prob=score[begin][split][B]*score[split][end][C]*P(A->BC)
    if prob > score[begin][end][A]
      score[begin][end][A] = prob
      back[begin][end][A] = new Triple(split,B,C)

```




CKY Parsing

A worked example

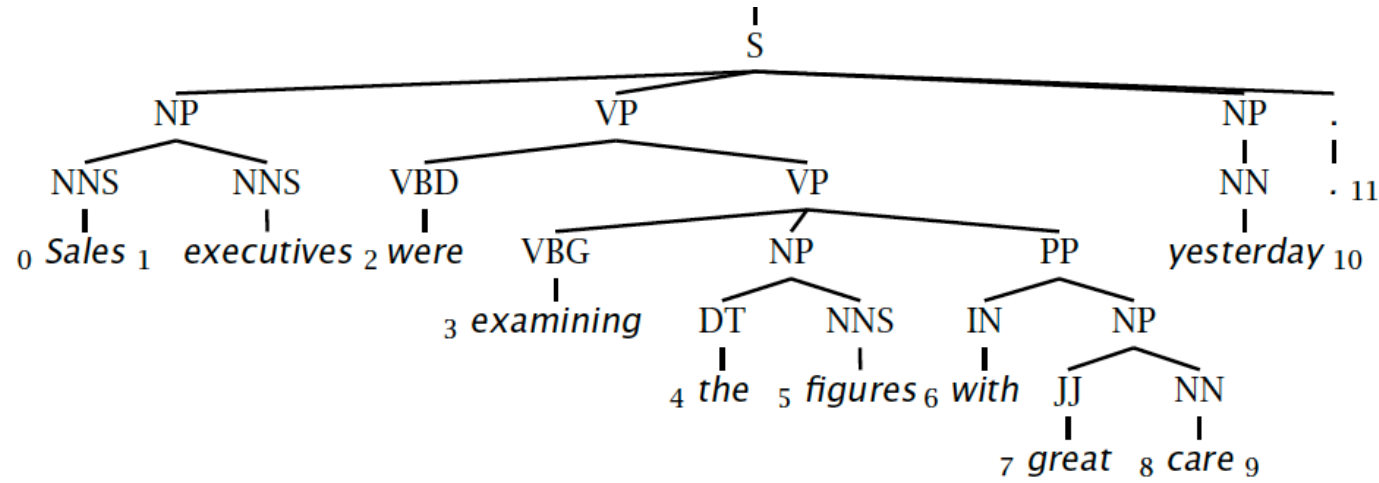


Constituency Parser Evaluation

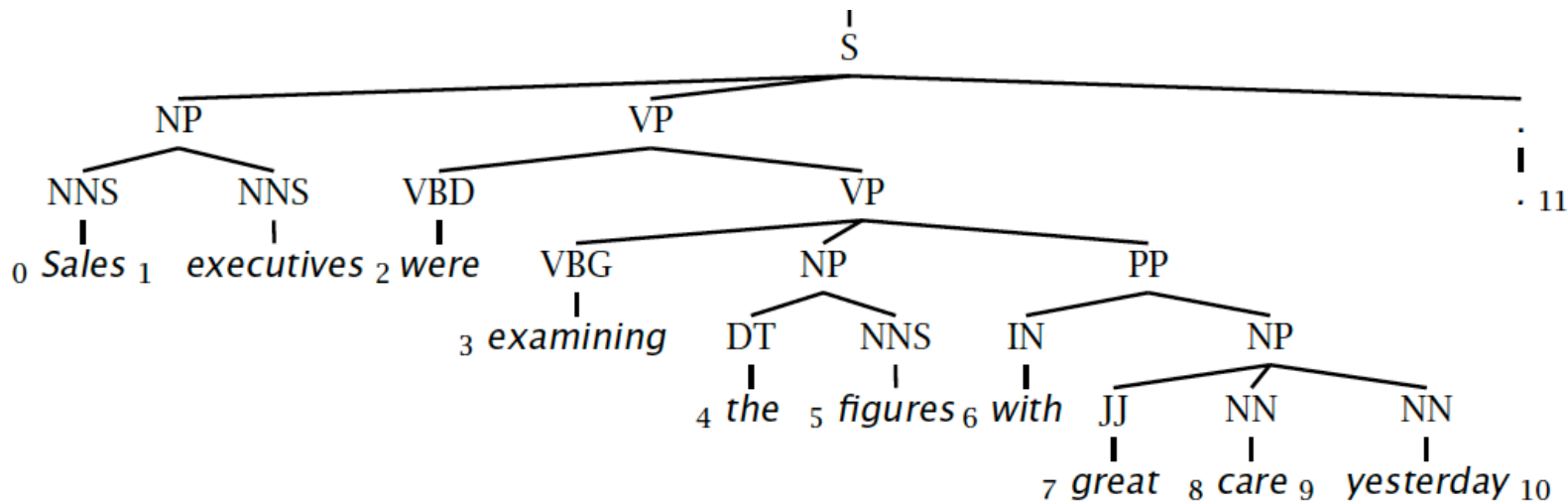


Evaluating constituency parsing

Gold standard brackets: S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), NP-(4:6), PP-(6:9), NP-(7,9), NP-(9:10)



Candidate brackets: S-(0:11), NP-(0:2), VP-(2:10), VP-(3:10), NP-(4:6), PP-(6:10), NP-(7,10)





Evaluating constituency parsing

Gold standard brackets:

S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), NP-(9:10)

Candidate brackets:

S-(0:11), NP-(0:2), VP-(2:10), VP-(3:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)

Labeled Precision	$3/7 = 42.9\%$
Labeled Recall	$3/8 = 37.5\%$
LP/LR F1	40.0%
Tagging Accuracy	$11/11 = 100.0\%$



How good are PCFGs?

- Penn WSJ parsing accuracy: about 73% LP/LR F1
- Robust
 - Usually admit everything, but with low probability
- Partial solution for grammar ambiguity
 - A PCFG gives some idea of the plausibility of a parse
 - But not so good because the independence assumptions are too strong
- Give a probabilistic language model
 - But in the simple case it performs worse than a trigram model
- The problem seems to be that PCFGs lack the lexicalization of a trigram model



Constituency Parser Evaluation