Syntactic Parsing and the Penn Treebank

JURAFSKY AND MARTIN CHAPTERS 11 AND 12

Based on slides from Sameer Singh, Dan Jurafsky, Noah Smith, Slav Petrov, and everyone else they copied from.

Source of Grammar?

Manual

Noam Chomsky

Write symbolic grammar (CFG or often richer) and lexicon

S $\rightarrow$ NP VP	NN $\rightarrow$ *interest*
NP $\rightarrow$ (DT) NN	NNS $\rightarrow$ *rates*
NP $\rightarrow$ NN NNS	NNS $\rightarrow$ *raises*
NP $\rightarrow$ NNP	VBP $\rightarrow$ *interest*
VP $\rightarrow$ V NP	VBZ $\rightarrow$ *rates*

Used grammar/proof systems to prove parses from words

Fed raises interest rates 0.5% in effort to control inflation

- Minimal grammar: 36 parses
- Simple 10 rule grammar: 592 parses
- Real-size broad-coverage grammar: millions of parses

Source of Grammar?

From data!

The Penn Treebank

Building a treebank seems a lot slower and less useful than building a grammar

But a treebank gives us many things

- Reusability of the labor
  - Many parsers, POS taggers, etc.
  - Valuable resource for linguistics
- Broad coverage
- Frequencies and distributional information
- A way to evaluate systems

[Marcus et al. 1993, *Computational Linguistics*]

```
( (S
    (NP-SBJ (DT The) (NN move))
    (VP (VBD followed)
      (NP
        (NP (DT a) (NN round))
        (PP (IN of)
          (NP
            (NP (JJ similar) (NNS increases))
            (PP (IN by)
              (NP (JJ other) (NNS lenders)))
            (PP (IN against)
              (NP (NNP Arizona) (JJ real) (NN estate) (NNS loans))))))
      (, ,)
      (S-ADV
        (NP-SBJ (-NONE- *))
        (VP (VBG reflecting)
          (NP
            (NP (DT a) (VBG continuing) (NN decline))
            (PP-LOC (IN in)
              (NP (DT that) (NN market)))))))
    (. .)))
```

```
( (S
    (NP-SBJ (DT The) (NN move))
    (VP (VBD followed)
      (NP
        (NP (DT a) (NN round))
        (PP (IN of)
          (NP
            (NP (JJ similar) (NNS increases))
            (PP (IN by)
              (NP (JJ other) (NNS lenders)))
            (PP (IN against)
              (NP (NNP Arizona) (JJ real) (NN estate) (NNS loans))))))
      (, ,)
      (S-ADV
        (NP-SBJ (-NONE- *))
        (VP (VBG reflecting)
          (NP
            (NP (DT a) (VBG continuing) (NN decline))
            (PP-LOC (IN in)
              (NP (DT that) (NN market)))))))
    (. .)))
```

Some of the rules, with counts

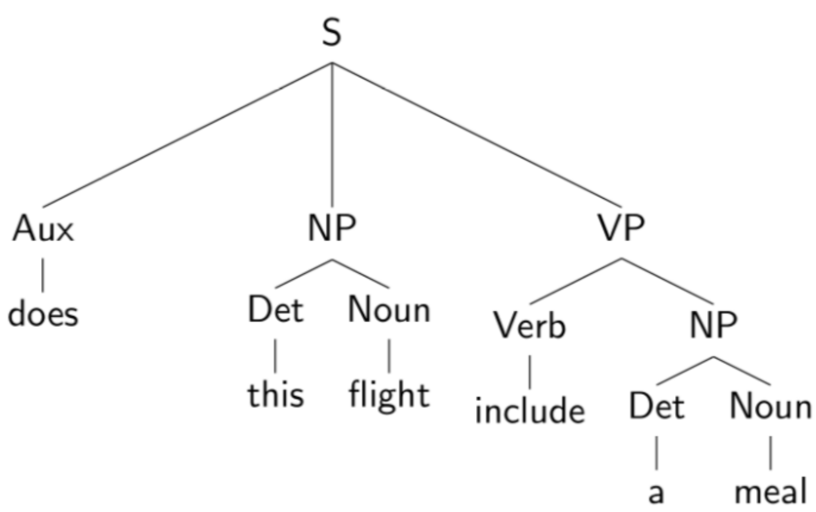
40717 PP → IN NP
33803 S → NP-SBJ VP
22513 NP-SBJ → -NONE-
21877 NP → NP PP
20740 NP → DT NN
14153 S → NP-SBJ VP .
12922 VP → TO VP
11881 PP-LOC → IN NP
11467 NP-SBJ → PRP
11378 NP → -NONE-
11291 NP → NN
...
989 VP → VBG S
985 NP-SBJ → NN
983 PP-MNR → IN NP
983 NP-SBJ → DT
969 VP → VBN VP

100 VP → VBD PP-PRD
100 PRN → : NP :
100 NP → DT JJS
100 NP-CLR → NN
99 NP-SBJ-1 → DT NNP
98 VP → VBN NP PP-DIR
98 VP → VBD PP-TMP
98 PP-TMP → VBG NP
97 VP → VBD ADVP-TMP VP
...
10 WHNP-1 → WRB JJ
10 VP → VP CC VP PP-TMP
10 VP → VP CC VP ADVP-MNR
10 VP → VBZ S , SBAR-ADV
10 VP → VBZ S ADVP-TMP

4500 rules for VP!

Evaluating Parses

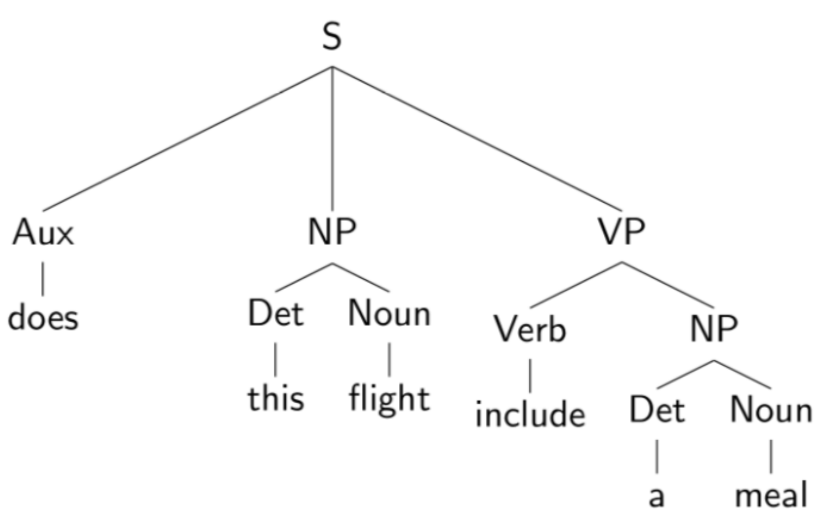
Each parse tree is represented by a list of tuples:



Use this to estimate precision/recall!

Evaluating Parses

Each parse tree is represented by a list of tuples: $\{\langle t_i, s_i, e_i \rangle\}$



$\langle S, 0, 6 \rangle$ $\langle Aux, 0, 1 \rangle$

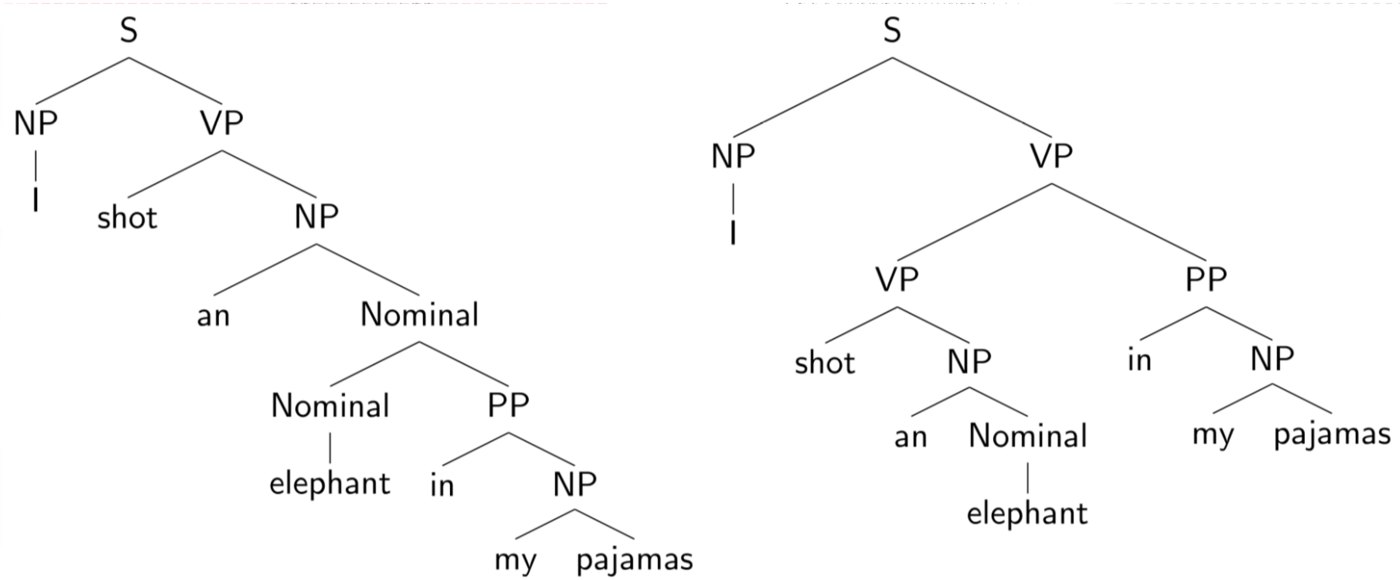
$\langle NP, 1, 3 \rangle$ $\langle DET, 1, 2 \rangle$

$\langle Noun, 2, 3 \rangle$ $\langle NP, 4, 6 \rangle$

$\langle VP, 3, 6 \rangle$ ...

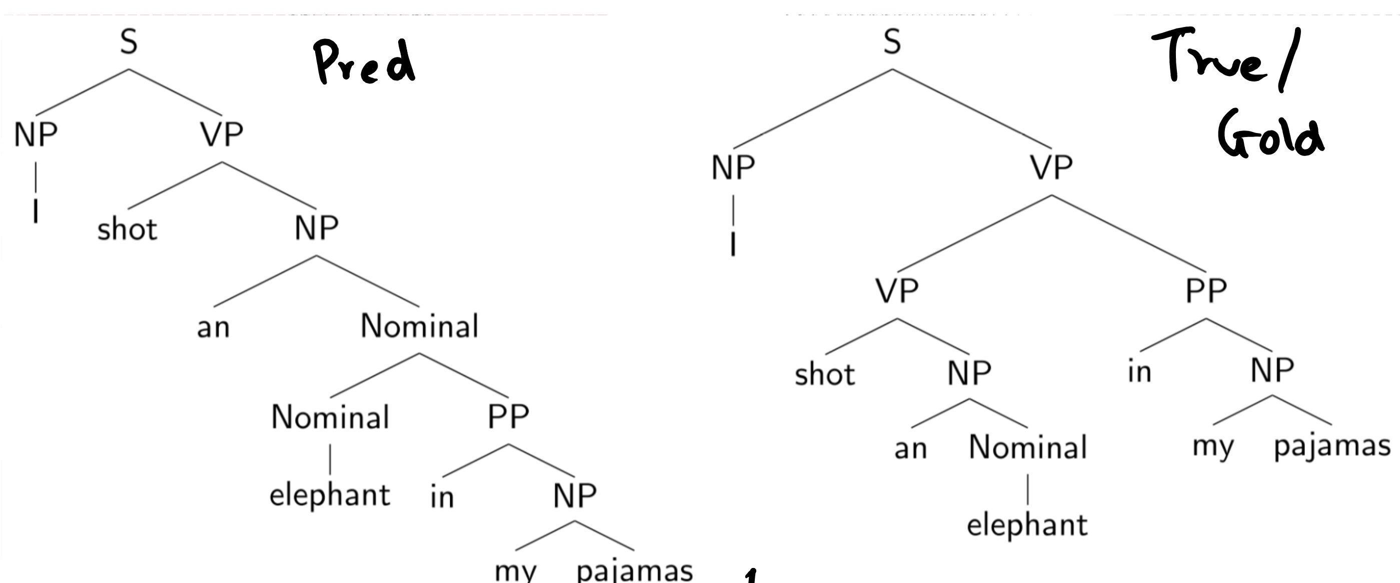
Use this to estimate precision/recall!

Evaluating Parses: Example



$$\underbrace{\left\langle \begin{array}{c}\langle \mathsf{NP},3,7\rangle,\\ \langle \mathsf{Nominal},4,7\rangle\end{array}\right\rangle}_{\text{only in left tree}}\underbrace{\left\langle \begin{array}{c}\langle \mathsf{S},1,7\rangle,\langle \mathsf{VP},2,7\rangle,\\ \langle \mathsf{PP},5,7\rangle,\langle \mathsf{NP},6,7\rangle\\ \langle \mathsf{Nominal},4,4\rangle\end{array}\right\rangle}_{\text{in both trees}}\underbrace{\left\langle \begin{array}{c}\langle \mathsf{VP},2,4\rangle,\\ \langle \mathsf{NP},3,4\rangle\end{array}\right\rangle}_{\text{only in right tree}}$$

Evaluating Parses: Example



fp tp fn

$$\underbrace{\left\langle \begin{matrix}\langle \mathsf{NP}, 3, 7\rangle, \\ \langle \mathsf{Nominal}, 4, 7\rangle\end{matrix} \right\rangle}_{\text{only in left tree}} \underbrace{\left\langle \begin{matrix}\langle \mathsf{S}, 1, 7\rangle, \langle \mathsf{VP}, 2, 7\rangle, \\ \langle \mathsf{PP}, 5, 7\rangle, \langle \mathsf{NP}, 6, 7\rangle \\ \langle \mathsf{Nominal}, 4, 4\rangle\end{matrix} \right\rangle}_{\text{in both trees}} \underbrace{\left\langle \begin{matrix}\langle \mathsf{VP}, 2, 4\rangle, \\ \langle \mathsf{NP}, 3, 4\rangle\end{matrix} \right\rangle}_{\text{only in right tree}}$$

Outline

Context Free Grammars

Parsing: CKY Algorithm

Extensions: Probabilistic and Lexicalized

Dependency Parsing

The Parsing Problem

Given sentence **x** and grammar **G**,

Recognition

Is sentence **x** in the grammar? If so, prove it.

"Proof" is a deduction, valid parse tree.

Parsing

Show one or more derivations for **x** in **G**.

Even with small grammars, brute force grows exponentially!

"Book that flight"

Left to Right?

The old man the boat.

The complex houses married and single soldiers and their families.

Garden Path Sentences

Left to Right?

S
NP VP —X
The old man the boat.
NP VP

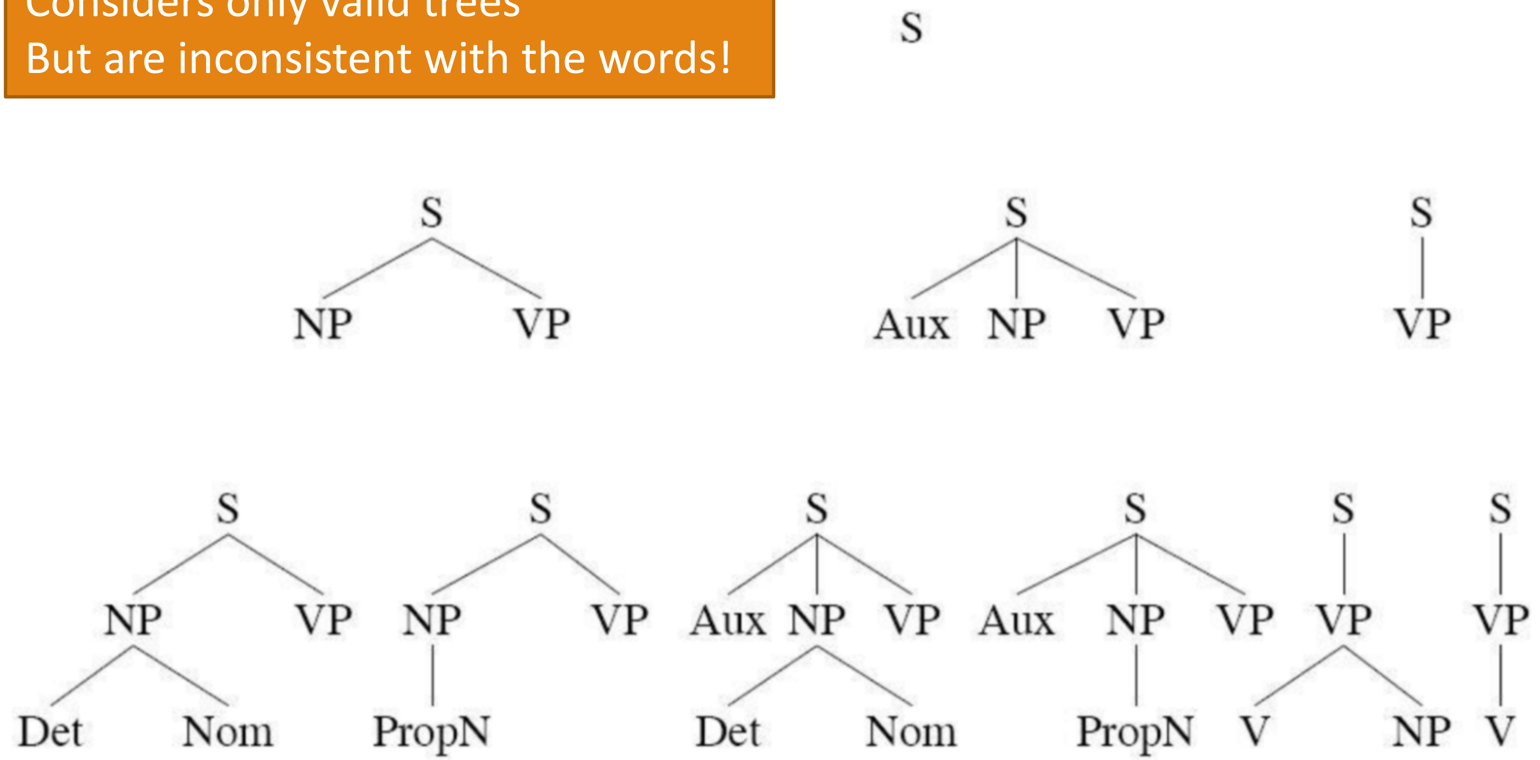
The complex houses married and single soldiers and their families.
NP Verb VP

Garden Path Sentences

Top Down Parsing

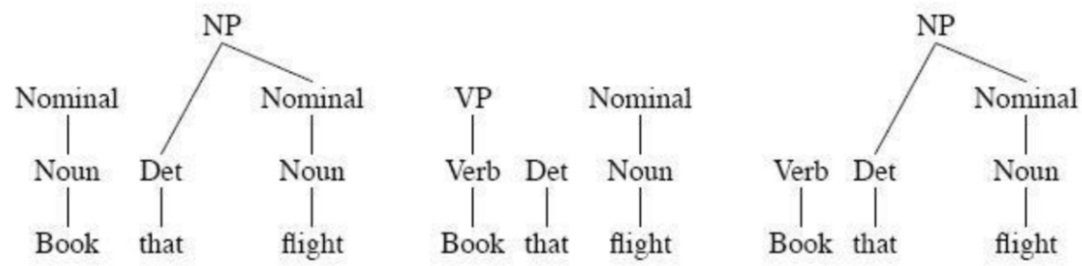
Considers only valid trees
But are inconsistent with the words!

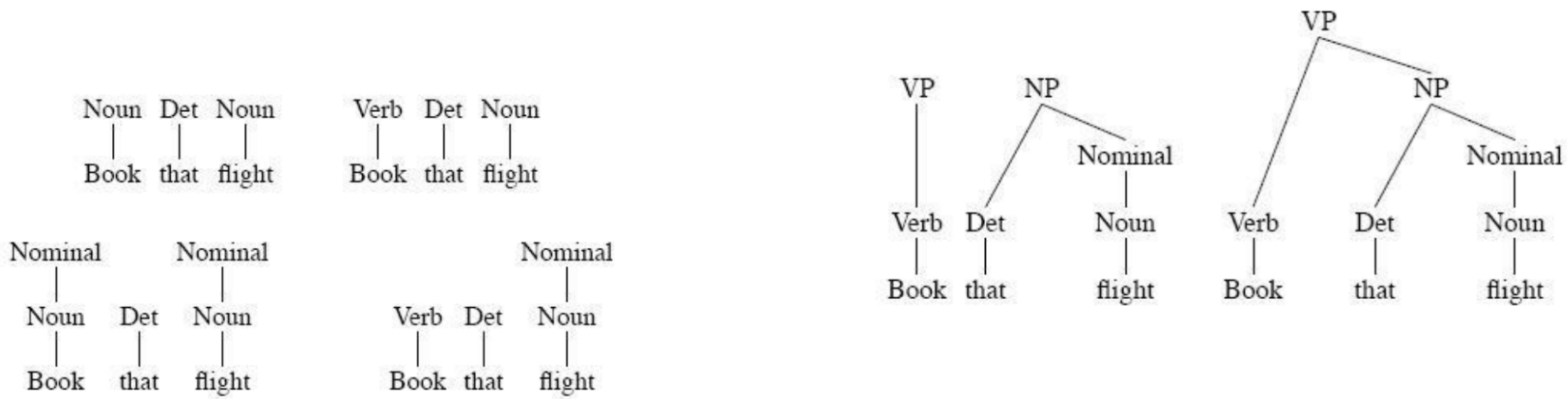
"Book that flight"



Bottom-up Parsing

"Book that flight"

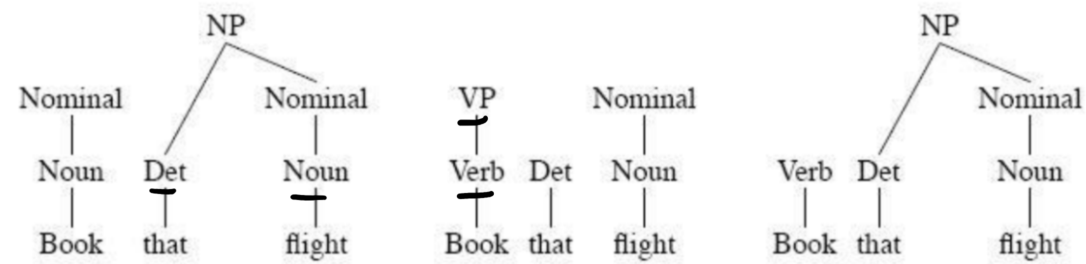


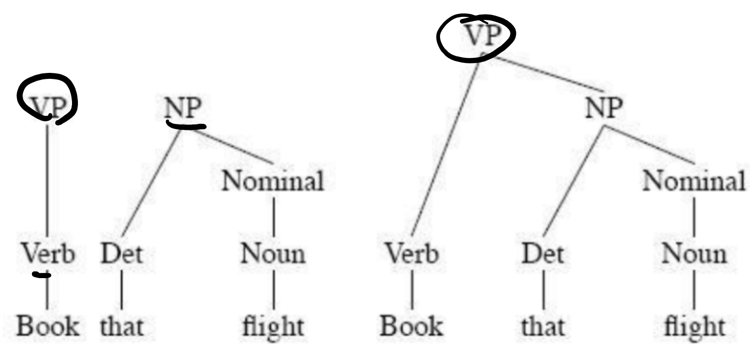


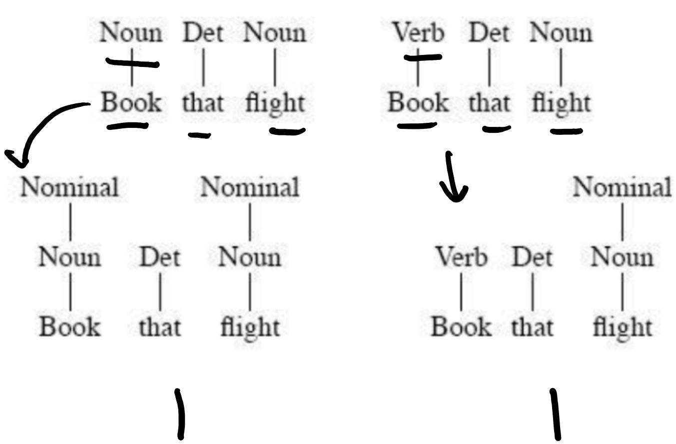
Builds only consistent trees
But most of them are invalid (don't go anywhere)!

Bottom-up Parsing

"Book that flight"







Builds only consistent trees
But most of them are invalid (don't go anywhere)!

Chomsky Normal Form

Context free grammar where all non-terminals to go:

- 2 non-terminals, or
- A single terminal

$A \rightarrow B\ C$ $\qquad D \rightarrow w$

Converting to CNF

Case 1

$$\begin{aligned} A &\rightarrow B \\ B &\rightarrow C\ D \\ B &\rightarrow w \end{aligned} \quad\Rightarrow\quad \begin{aligned} A &\rightarrow C\ D \\ A &\rightarrow w \end{aligned}$$

Case 2

$$A \rightarrow B\ C\ D\ E \quad\Rightarrow\quad \begin{aligned} A &\rightarrow X\ E \\ X &\rightarrow Y\ D \\ Y &\rightarrow B\ C \end{aligned}$$

Original Grammar	Chomsky Normal Form
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

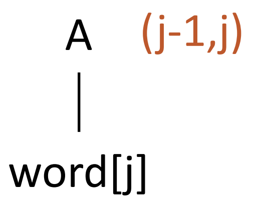
Dynamic Programming

table[i,j] = Set of all valid non-terminals for the constituent span (i,j)

Base case

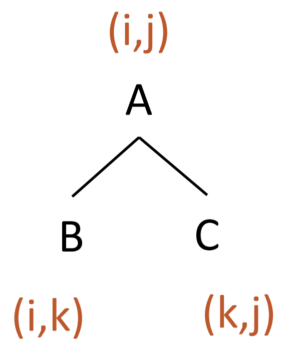
Rule: $A \rightarrow word[j]$

A should be in table[j-1,j]



Recursion

Rule: $A \rightarrow B\ C$



If you find a k such that

B is in table[i,k], and

C is in table[k,j], then A should be in table[i,j]

Dynamic Programming

$S \in table[0,n]$

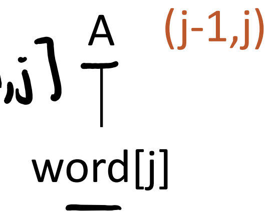
table[i,j] = Set of all valid non-terminals for the constituent span (i,j)

Base case

Rule: *A → word[j]* ←

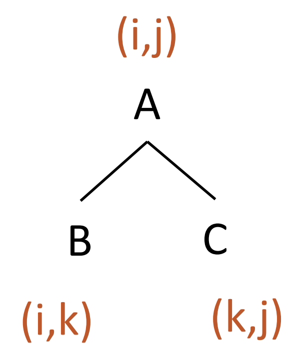
A should be in table[j-1,j]

$A \in table[j-1,j]$



Recursion

Rule: *A → B C*



If you find a k such that

B is in table[i,k], and

C is in table[k,j], then A should be in table[i,j]

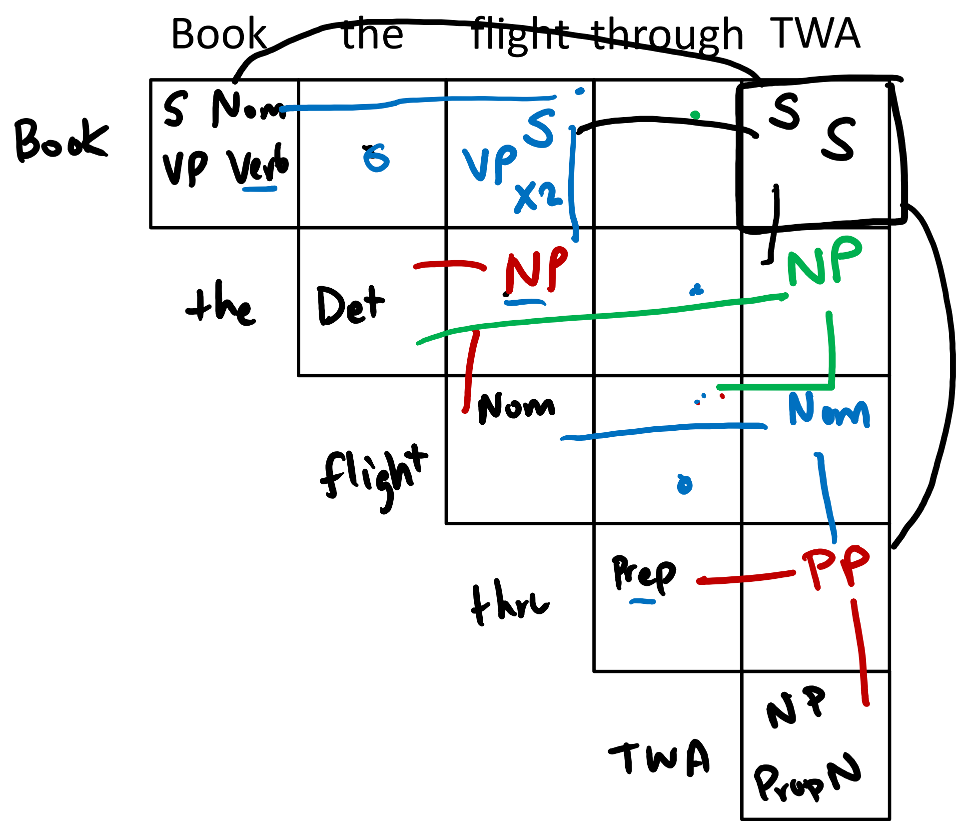
CKY Algorithm

$S \rightarrow NP\ VP$
$S \rightarrow X1\ VP$
$X1 \rightarrow Aux\ NP$
$S \rightarrow book \mid include \mid prefer$
$S \rightarrow Verb\ NP$
$S \rightarrow X2\ PP$
$S \rightarrow Verb\ PP$
$S \rightarrow VP\ PP$
$NP \rightarrow I \mid she \mid me$
$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$
$VP \rightarrow X2\ PP$
$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$

Book	the	flight	through	TWA

CKY Algorithm

$S \rightarrow NP\ VP$
$S \rightarrow X1\ VP$
$X1 \rightarrow Aux\ NP$
$S \rightarrow book \mid include \mid prefer$
$S \rightarrow Verb\ NP$
$S \rightarrow X2\ PP$
$S \rightarrow Verb\ PP$
$S \rightarrow VP\ PP$
$NP \rightarrow I \mid she \mid me$
$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$
$VP \rightarrow X2\ PP$
$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$

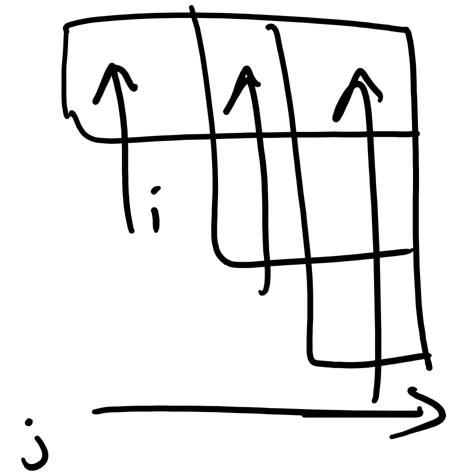


CKY Algorithm

function CKY-PARSE(*words, grammar*) **returns** *table*

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  for all $\{A \mid A \rightarrow words[j] \in grammar\}$
    $table[j-1, j] \leftarrow table[j-1, j] \cup A$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      for all $\{A \mid A \rightarrow BC \in grammar$ **and** $B \in table[i,k]$ **and** $C \in table[k,j]\}$
        $table[i,j] \leftarrow table[i,j] \cup A$

CKY Algorithm



function CKY-PARSE(*words, grammar*) **returns** *table*

n
  for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
    for all $\{A \mid A \rightarrow words[j] \in grammar\}$
      $table[j-1, j] \leftarrow table[j-1, j] \cup A$

n
    for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**

n
      for $k \leftarrow i+1$ **to** $j-1$ **do**

|R|
        for all $\{A \mid A \rightarrow BC \in grammar$ **and** $B \in table[i,k]$ **and** $C \in table[k,j]\}$
          $table[i,j] \leftarrow table[i,j] \cup A$

CKY Algorithm: Complexity

$|N|$: Number of non-terminals

$|R|$: Number of rules

n: Number of tokens in the sentence

Memory

Time

CKY Algorithm: Complexity

$|N|$: Number of non-terminals

$|R|$: Number of rules

n: Number of tokens in the sentence

Memory $O(n^2 |N|)$

Time $O(n^3 |R|)$

Outline

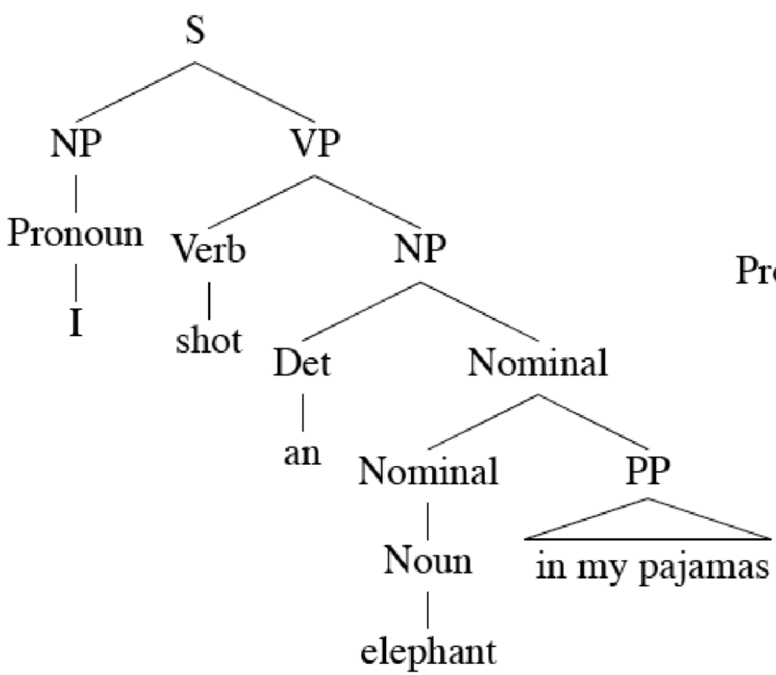
Parsing: CKY Algorithm

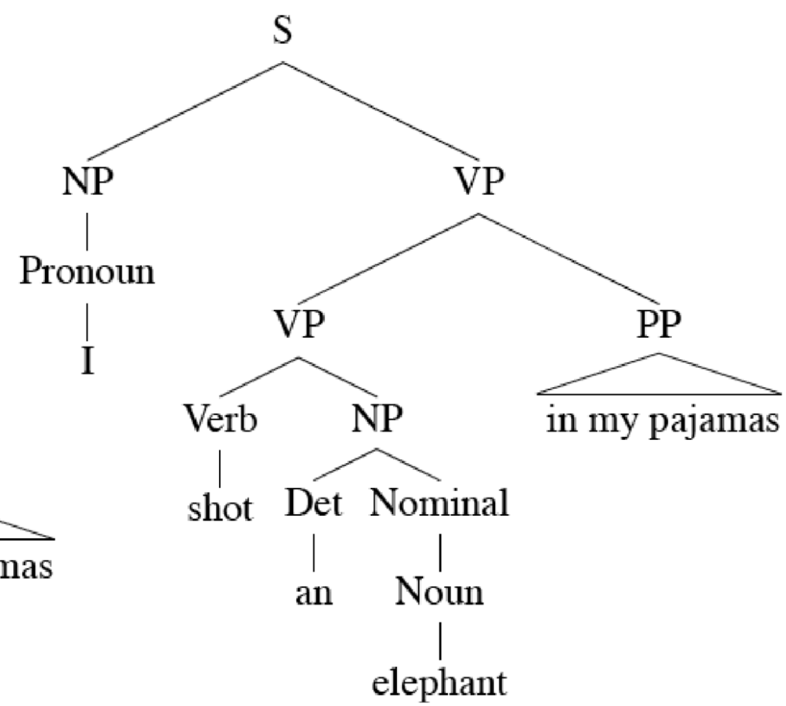
Extensions: Probabilistic and Lexicalized

Dependency Parsing

Ambiguity: Which parse?

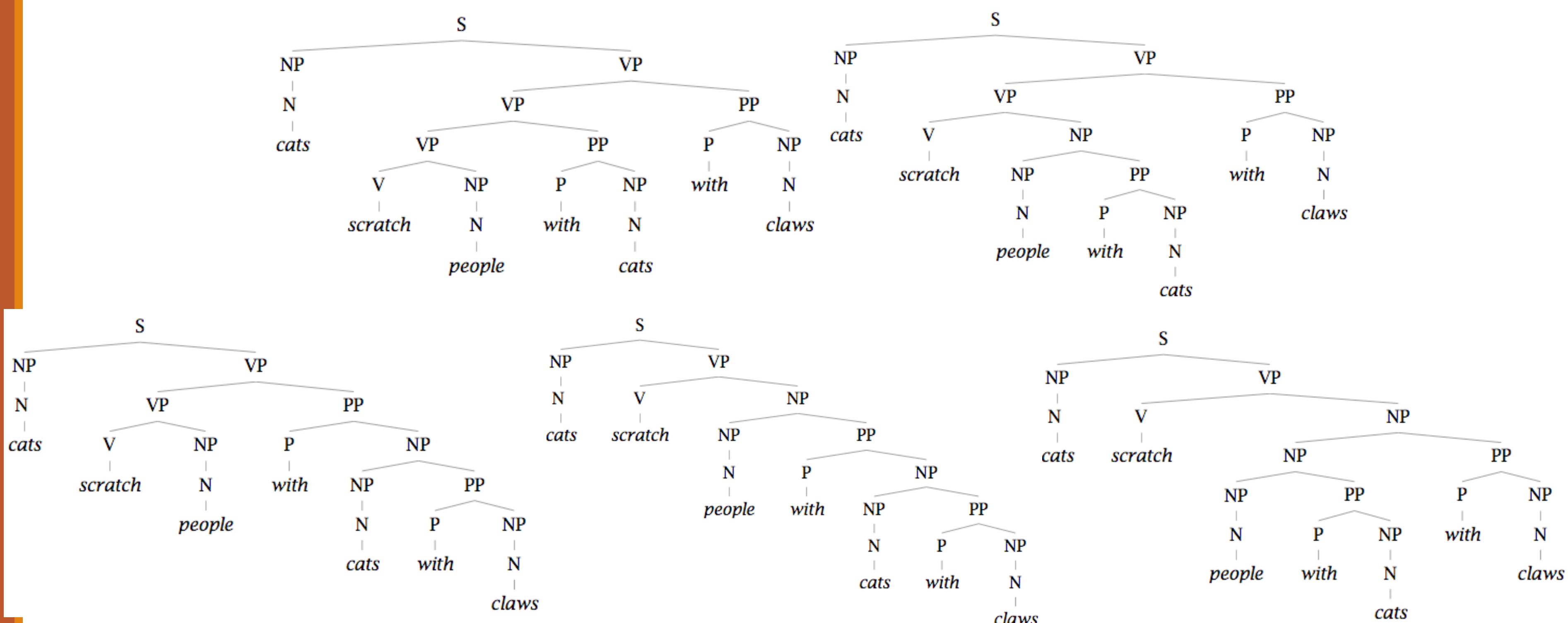
I shot an elephant in my pajamas.





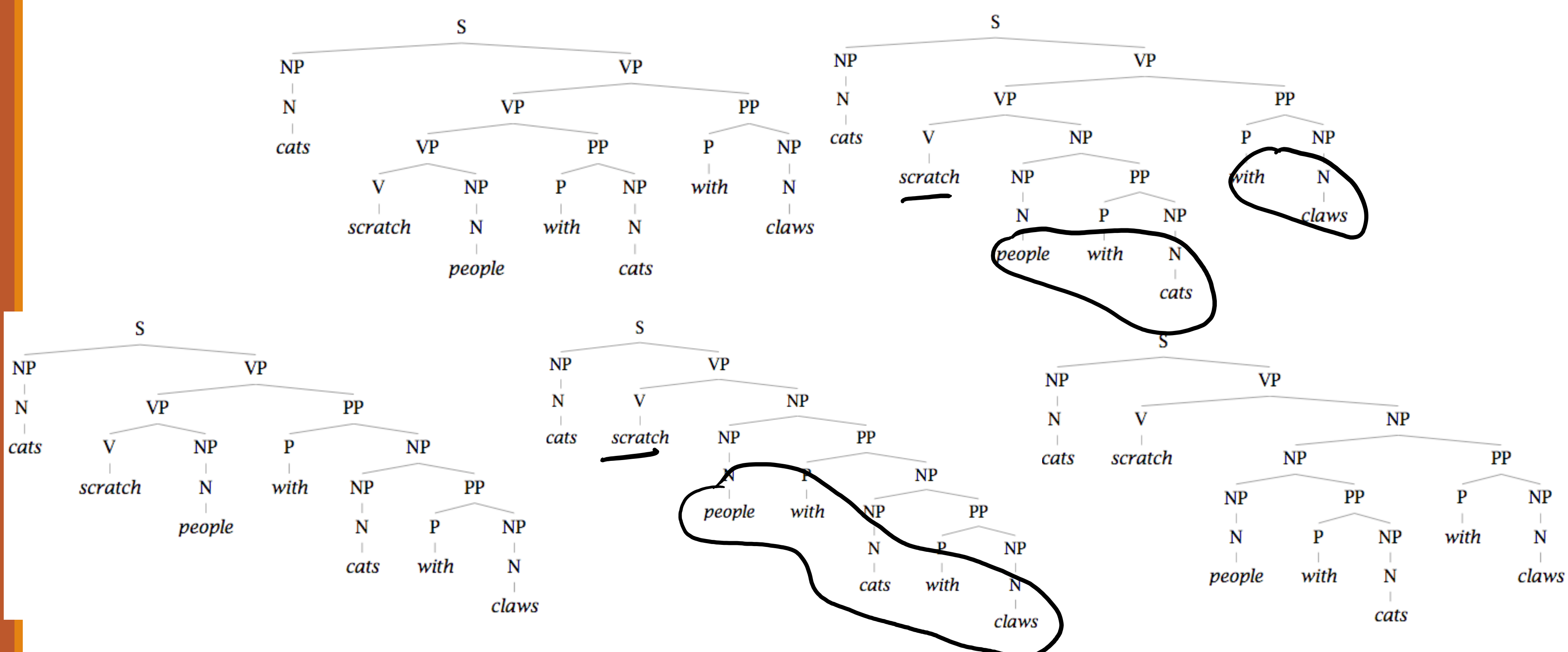
Finding the *Best* Parse Tree

Cats scratch people with cats with claws.



Finding the *Best* Parse Tree

Cats scratch people with cats with claws.



Probabilistic CFGs

Same as a regular context-free grammar:

- Terminal, non-terminals, and rules
- Additionally, attach a probability to each rule!

Rule: $A \rightarrow B\ C$

Probability: $P(A \rightarrow B\ C \mid A)$

Compute the probability of a parse tree:

Probabilistic CFGs

Same as a regular context-free grammar:

- Terminal, non-terminals, and rules
- Additionally, attach a probability to each rule!

Rule: $A \rightarrow B\ C$ Probability: P($A \rightarrow B\ C \mid A$)

Compute the probability of a parse tree: $\prod_{A \to BC \in T} P(A \to BC \mid A)$